

Documented Code For glossaries v4.08

Nicola L.C. Talbot

Dickimaw Books

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

2014-07-30

This is the documented code for the glossaries package. This bundle comes with the following documentation:

[glossariesbegin.pdf](#) If you are a complete beginner, start with “The glossaries package: a guide for beginners”.

[glossary2glossaries.pdf](#) If you are moving over from the obsolete glossary package, read “Upgrading from the glossary package to the glossaries package”.

[glossaries-user.pdf](#) For the main user guide, read “glossaries.sty v4.08: \TeX 2e Package to Assist Generating Glossaries”.

[mfirstuc-manual.pdf](#) The commands provided by the mfirstuc package are briefly described in “mfirstuc.sty: uppercasing first letter”.

[glossaries-code.pdf](#) This document is for advanced users wishing to know more about the inner workings of the glossaries package.

INSTALL Installation instructions.

CHANGES Change log.

README Package summary.

Contents

1 Main Package Code	3
1.1 Package Definition	3
1.2 Package Options	5
1.3 Default values	29
1.4 Xindy	39
1.5 Loops and conditionals	48
1.6 Defining new glossaries	53
1.7 Defining new entries	57
1.8 Resetting and unsetting entry flags	78
1.9 Loading files containing glossary entries	80
1.10 Using glossary entries in the text	81
1.10.1 Links to glossary entries	91
1.10.2 Displaying entry details without adding information to the glossary	132
1.11 Adding an entry to the glossary without generating text	139
1.12 Creating associated files	141
1.13 Writing information to associated files	156
1.14 Glossary Entry Cross-References	161
1.15 Displaying the glossary	163
1.16 Acronyms	192
1.17 Predefined acronym styles	197
1.18 Predefined Glossary Styles	228
1.19 Debugging Commands	228
1.20 Compatibility with version 2.07 and below	234
2 Prefix Support (glossaries-prefix Code)	234
3 Mfirstuc Documented Code	240
4 Mfirstuc-english Documented Code	243
5 Glossary Styles	244
5.1 Glossary hyper-navigation definitions (glossary-hypernav package)	244
5.2 In-line Style (glossary-inline.sty)	246
5.3 List Style (glossary-list.sty)	248
5.4 Glossary Styles using longtable (the glossary-long package)	251
5.5 Glossary Styles using longtable (the glossary-longragged package)	258
5.6 Glossary Styles using multicol (glossary-mcols.sty)	263
5.7 Glossary Styles using supertabular environment (glossary-super package)	267
5.8 Glossary Styles using supertabular environment (glossary-superragged package)	274
5.9 Tree Styles (glossary-tree.sty)	280

6 glossaries-compatible-207	287
7 Accessibility Support (glossaries-accsupp Code)	307
7.1 Defining Replacement Text	308
7.2 Accessing Replacement Text	312
7.3 Displaying the Glossary	327
7.4 Acronyms	328
7.5 Debugging Commands	342
8 Multi-Lingual Support	344
8.1 Babel Captions	344
8.2 Polyglossia Captions	350
8.3 Brazilian Dictionary	353
8.4 Danish Dictionary	353
8.5 Dutch Dictionary	354
8.6 English Dictionary	354
8.7 French Dictionary	354
8.8 German Dictionary	354
8.9 Irish Dictionary	355
8.10 Italian Dictionary	355
8.11 Magyar Dictionary	355
8.12 Polish Dictionary	356
8.13 Serbian Dictionary	356
8.14 Spanish Dictionary	356
Glossary	356
Change History	357
Index	380

1 Main Package Code

1.1 Package Definition

This package requires $\text{\LaTeX}2_{\epsilon}$.

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries}[2014/07/30 v4.08 (NLCT)]
```

Required packages:

```
3 \RequirePackage{ifthen}
4 \RequirePackage{xkeyval}[2006/11/18]
5 \RequirePackage{mfirstuc}
```

The textcase package has much better case changing handling, so use `\MakeTextUppercase` instead of `\MakeUppercase`

```
6 \RequirePackage{textcase}
```

```
7 \renewcommand*{\mfirstucMakeUppercase}{\MakeTextUppercase}%
```

```
8 \RequirePackage{xfor}
```

```
9 \RequirePackage{datatool-base}
```

Need to use `\new@ifnextchar` instead of `\@ifnextchar` in commands that have a final optional argument (such as `\gls`) so require `.` Thanks to Morten Høgholm for suggesting this. (This has replaced using the `xspace` package.)

```
10 \RequirePackage{amsgen}
```

As from v3.0, now loading `etoolbox`:

```
11 \RequirePackage{etoolbox}
```

Check if doc has been loaded.

```
\if@gls@docloaded
```

```
12 \newif\if@gls@docloaded
```

```
13 \@ifpackageloaded{doc}%
```

```
14 {%
```

```
15   \@gls@docloadedtrue
```

```
16 }%
```

```
17 {%
```

```
18   \@ifclassloaded{nlctdoc}{\@gls@docloadedtrue}{\@gls@docloadedfalse}%
```

```
19 }
```

```
20 \if@gls@docloaded
```

`\doc` has been loaded, so some modifications need to be made to ensure both packages can work together.

```
\glsorg@glossary First, save the original behaviour of \glossary
```

```
21 \newcommand{\glsorg@glossary}{%
```

```
22   \@bsphack
```

```
23   \begingroup
```

```
24     \@sanitize \endgroup\@esphack
```

```
25 }
```

```
\glsorg@wrglossary
```

```
26 \newcommand{\glsorg@wrglossary}[1]{%
```

```
27   \protected@write\@glossaryfile{}{%
```

```
28     \string \glossaryentry{#1}{\thepage}}%
```

```
29   \endgroup
```

```
30   \@esphack
```

```
31 }
```

```
32 \renewcommand*{\RecordChanges}{%
```

```
33   \newwrite\@glossaryfile
```

```
34   \immediate\openout\@glossaryfile=\jobname.glo
```

```
35   \def\glsorg@glossary{\@bsphack\begingroup\@sanitize\glsorg@wrglossary}%
```

```
36   \typeout{Writing glossary file \jobname .glo}%
```

```
37 }
```

`\changes` Now we need to redefine `\changes` so that it uses the original definition of `\glossary`.

```
38 \let\glsorg@changes\changes
39 \renewcommand{\changes}[3]{%
40   \begingroup
41     \let\glossary\glsorg@glossary
42     \glsorg@changes{#1}{#2}{#3}%
43   \endgroup
44 }
```

`\PrintChanges` needs to use doc's version of `theglossary`, so save that.

`\glsorg@theglossary`

```
45 \let\glsorg@theglossary\theglossary
```

`\glsorg@endtheglossary`

```
46 \let\glsorg@endtheglossary\endtheglossary
```

`\PrintChanges` Now redefine `\PrintChanges` so that it uses the original `theglossary` environment.

```
47 \let\glsorg@PrintChanges\PrintChanges
48 \renewcommand{\PrintChanges}{%
49   \begingroup
50     \let\theglossary\glsorg@theglossary
51     \let\endtheglossary\glsorg@endtheglossary
52     \glsorg@PrintChanges
53   \endgroup
54 }
```

End of doc stuff.

```
55 \fi
```

1.2 Package Options

`toc` The `toc` package option will add the glossaries to the table of contents. This is a boolean key, if the value is omitted it is taken to be true.

```
56 \define@boolkey{glossaries.sty}[gls]{toc}[true]{}%
```

`numberline` The `numberline` package option adds `\numberline` to `\addcontentsline`. Note that this option only has an effect if used in with `toc=true`.

```
57 \define@boolkey{glossaries.sty}[gls]{numberline}[true]{}%
```

`\@@glossarysec` The sectional unit used to start the glossary is stored in `\@@glossarysec`. If chapters are defined, this is initialised to `chapter`, otherwise it is initialised to `section`.

```
58 \ifcsundef{chapter}%
59   {\newcommand*\@@glossarysec{section}}%
60   {\newcommand*\@@glossarysec{chapter}}
```

`section` The `section` key can be used to set the sectional unit. If no unit is specified, use `section` as the default. The starred form of the named sectional unit will be used. If you want some other way to start the glossary section (e.g. a numbered section) you will have to redefine `\glossarysection`.

```
61 \define@choicekey{glossaries.sty}{section}{part,chapter,section,%
62 subsection,subsubsection,paragraph,subparagraph}[section]{%
63   \renewcommand*{\@@glossarysec}{#1}}
```

Determine whether or not to use numbered sections.

`\@@glossarysecstar`

```
64 \newcommand*{\@@glossarysecstar}{*}
```

`\@@glossaryseclabel`

```
65 \newcommand*{\@@glossaryseclabel}{}
```

`\glsautoprefix` Prefix to add before label if automatically generated:

```
66 \newcommand*{\glsautoprefix}{}
```

`numberedsection`

```
67 \define@choicekey{glossaries.sty}{numberedsection}[\val\nr]{%
68 false,nolabel,autolabel,nameref}[nolabel]{%
69   \ifcase\nr\relax
70     \renewcommand*{\@@glossarysecstar}{*}%
71     \renewcommand*{\@@glossaryseclabel}{}%
72   \or
73     \renewcommand*{\@@glossarysecstar}{}%
74     \renewcommand*{\@@glossaryseclabel}{}%
75   \or
76     \renewcommand*{\@@glossarysecstar}{}%
77     \renewcommand*{\@@glossaryseclabel}{}%
78     \label{\glsautoprefix@glo@type}%
79   \or
80     \renewcommand*{\@@glossarysecstar}{*}%
81     \renewcommand*{\@@glossaryseclabel}{}%
82     \protected@edef\@currentlabelname{\glossarytoctitle}%
83     \label{\glsautoprefix@glo@type}%
84   \fi
85 }
```

The default glossary style is stored in `\@glossary@default@style`. This is initialised to `list`. (The `list` style is defined in the accompanying package described in [subsection 1.18](#).)

`\@glossary@default@style`

```
86 \newcommand*{\@glossary@default@style}{list}
```

style The default glossary style can be changed using the style package option. The value can be the name of any defined glossary style. The glossary style is set at the beginning of the document, so you can still use the style key to set a style that is defined in another package. This package comes with some predefined styles that are defined in [subsection 1.18](#).

```
87 \define@key{glossaries.sty}{style}{%
88   \renewcommand*{\@glossary@default@style}{#1}%
89 }
```

Each `\DeclareOptionX` needs a corresponding `\DeclareOption` so that it can be passed as a document class option, so define a command that will implement both.

`\@gls@declareoption`

```
90 \newcommand*{\@gls@declareoption}[2]{%
91   \DeclareOptionX{#1}{#2}%
92   \DeclareOption{#1}{#2}%
93 }
```

Each entry within a given glossary will have an associated number list. By default, this refers to the page numbers on which that entry has been used, but it can also refer to any counter used in the document (such as the section or equation counters). The default number list format displays the number list “as is”:

`\glossaryentrynumbers`

```
94 \newcommand*{\glossaryentrynumbers}[1]{#1\gls@save@numberlist{#1}}
```

nonumberlist Note that the entire number list for a given entry will be passed to `\glossaryentrynumbers` so any font changes will also be applied to the delimiters. The `nonumberlist` package option suppresses the number lists (this simply redefines `\glossaryentrynumbers` to ignore its argument).

```
95 \@gls@declareoption{nonumberlist}{%
96   \renewcommand*{\glossaryentrynumbers}[1]{\gls@save@numberlist{#1}}%
97 }
```

savenumberlist Provide means to store the number list for entries.

```
98 \define@boolkey{glossaries.sty}[gls]{savenumberlist}[true]{}
99 \glssavenumberlistfalse
```

`\@glo@seeautonumberlist`

```
100 \newcommand*\@glo@seeautonumberlist{}
```

seeautonumberlist Automatically activates number list for entries containing the see key.

```
101 \@gls@declareoption{seeautonumberlist}{%
102   \renewcommand*{\@glo@seeautonumberlist}{%
103     \def\@glo@prefix{\glsnextpages}%
104   }%
105 }
```

```

\@gls@loadlong
106 \newcommand*{\@gls@loadlong}{\RequirePackage{glossary-long}}

nolong This option prevents from being loaded. This means that the glossary styles
that use the longtable environment will not be available. This option is pro-
vided to reduce overhead caused by loading unrequired packages.
107 \@gls@declareoption{nolong}{\renewcommand*{\@gls@loadlong}{}}

\@gls@loadsuper The package isn't loaded if isn't installed.
108 \IfFileExists{supertabular.sty}{%
109 \newcommand*{\@gls@loadsuper}{\RequirePackage{glossary-super}}}{%
110 \newcommand*{\@gls@loadsuper}{}}

nosuper This option prevents from being loaded. This means that the glossary styles
that use the supertabular environment will not be available. This option is pro-
vided to reduce overhead caused by loading unrequired packages.
111 \@gls@declareoption{nosuper}{\renewcommand*{\@gls@loadsuper}{}}

\@gls@loadlist
112 \newcommand*{\@gls@loadlist}{\RequirePackage{glossary-list}}

nolist This option prevents from being loaded (to reduce overheads if required). Nat-
urally, the styles defined in will not be available if this option is used.
113 \@gls@declareoption{nolist}{\renewcommand*{\@gls@loadlist}{}}

\@gls@loadtree
114 \newcommand*{\@gls@loadtree}{\RequirePackage{glossary-tree}}

notree This option prevents from being loaded (to reduce overheads if required). Nat-
urally, the styles defined in will not be available if this option is used.
115 \@gls@declareoption{notree}{\renewcommand*{\@gls@loadtree}{}}

nostyles Provide an option to suppress all the predefined styles (in the event that the
user has custom styles that are not dependent on the predefined styles).
116 \@gls@declareoption{nostyles}{%
117 \renewcommand*{\@gls@loadlong}{}%
118 \renewcommand*{\@gls@loadsuper}{}%
119 \renewcommand*{\@gls@loadlist}{}%
120 \renewcommand*{\@gls@loadtree}{}%
121 \let\@glossary@default@style\relax
122 }

\glspostdescription The description terminator is given by \glspostdescription (except for the
3 and 4 column styles). This is a full stop by default. The spacefactor is ad-
justed in case the description ends with an upper case letter. (Patch provided
by Michael Pock.)

```



```

123 \newcommand*{\glspostdescription}{%
124   \ifglsnopostdot\else.\spacefactor\sfcode'\. \fi
125 }

nopostdot   Boolean option to suppress post description dot
126 \define@boolkey{glossaries.sty}[gls]{nopostdot}[true]{}
127 \glsnopostdotfalse

nogroupskip Boolean option to suppress vertical space between groups in the pre-defined
styles.
128 \define@boolkey{glossaries.sty}[gls]{nogroupskip}[true]{}
129 \glsnogroupskipfalse

ucmark      Boolean option to determine whether or not to use use upper case in definition
of \glsglossarymark
130 \define@boolkey{glossaries.sty}[gls]{ucmark}[true]{}

131 \@ifclassloaded{memoir}
132 {%
133   \glsucmarktrue
134 }%
135 {%
136   \glsucmarkfalse
137 }

entrycounter Defines a counter that can be used in the standard glossary styles to number
each (main) entry. If true, this will define a counter called glossaryentry.
138 \define@boolkey{glossaries.sty}[gls]{entrycounter}[true]{}
139 \glsentrycounterfalse

entrycounterwithin This option can be used to set a parent counter for glossaryentry. This option
automatically sets entrycounter=true.
140 \define@key{glossaries.sty}{counterwithin}{%
141   \renewcommand*{\@gls@counterwithin}{#1}%
142   \glsentrycountertrue
143 }

\@gls@counterwithin The default value is no parent counter:
144 \newcommand*{\@gls@counterwithin}{}

subentrycounter Define a counter that can be used in the standard glossary styles to number
each level 1 entry. If true, this will define a counter called glossarysubentry.
145 \define@boolkey{glossaries.sty}[gls]{subentrycounter}[true]{}
146 \glssubentrycounterfalse

lo@default@sorttype Initialise default sort for \printnoidxglossary
147 \newcommand*{\@glo@default@sorttype}{standard}

```

sort Define the sort method: sort=standard (default), sort=def (order of definition) or sort=use (order of use).

```
148 \define@choicekey{glossaries.sty}{sort}{standard,def,use}{%
149   \renewcommand*{\@glo@default@sorttype}{#1}%
150   \csname @gls@setupsort@#1\endcsname
151 }
```

`\glsprestandardsort` `\glsprestandardsort{<sort cs>}{<type>}{<label>}`

Allow user to hook into sort mechanism. The first argument *<sort cs>* is the temporary control sequence containing the sort value before it has been sanitized and had `makeindex/xindy` special characters escaped.

```
152 \newcommand*{\glsprestandardsort}[3]{%
153   \glsdosanitizesort
154 }
```

`@setupsort@standard` Set up the macros for default sorting.

```
155 \newcommand*{\@gls@setupsort@standard}{%
```

Store entry information when it's defined.

```
156   \def\do@glo@storeentry{\@glo@storeentry}%
```

No count register required for standard sort.

```
157   \def\@gls@defsortcount##1{%
```

Sort according to sort key (`\@glo@sort`) if provided otherwise sort according to the entry's name (`\@glo@name`). (First argument glossary type, second argument entry label.)

```
158   \def\@gls@defsort##1##2{%
```

```
159     \ifx\@glo@sort\@glsdefaultsort
```

```
160       \let\@glo@sort\@glo@name
```

```
161     \fi
```

```
162     \let\glsdosanitizesort\@gls@sanitizesort
```

```
163     \glsprestandardsort{\@glo@sort}{##1}{##2}%
```

```
164     \expandafter\protected@xdef\csname glo@##2@sort\endcsname{\@glo@sort}%
```

```
165   }%
```

Don't need to do anything when the entry is used.

```
166   \def\@gls@setsort##1{%
```

```
167 }
```

Set standard sort as the default:

```
168 \@gls@setupsort@standard
```

`\glssortnumberfmt` Format the number used as the sort key by sort=def and sort=use. Defaults to six digit numbering.

```
169 \newcommand*\glssortnumberfmt[1]{%
```

```

170 \ifnum#1<100000 0\fi
171 \ifnum#1<10000 0\fi
172 \ifnum#1<1000 0\fi
173 \ifnum#1<100 0\fi
174 \ifnum#1<10 0\fi
175 \number#1%
176 }

```

`\@gls@setupsort@def` Set up the macros for order of definition sorting.

```
177 \newcommand*{\@gls@setupsort@def}{%
```

Store entry information when it's defined.

```
178 \def\do@glo@storeentry{\@glo@storeentry}%
```

Defined count register associated with the glossary.

```
179 \def\@gls@defsortcount##1{%
```

```
180 \expandafter\global
```

```
181 \expandafter\newcount\csname glossary@##1@sortcount\endcsname
```

```
182 }%
```

Increment count register associated with the glossary and use as the sort key.

```
183 \def\@gls@defsort##1##2{%
```

```
184 \expandafter\global\expandafter
```

```
185 \advance\csname glossary@##1@sortcount\endcsname by 1\relax
```

```
186 \expandafter\protected@xdef\csname glo@##2@sort\endcsname{%
```

```
187 \expandafter\glssortnumberfmt
```

```
188 {\csname glossary@##1@sortcount\endcsname}}%
```

```
189 }%
```

Don't need to do anything when the entry is used.

```
190 \def\@gls@setsort##1{%
```

```
191 }
```

`\@gls@setupsort@use` Set up the macros for order of use sorting.

```
192 \newcommand*{\@gls@setupsort@use}{%
```

Don't store entry information when it's defined.

```
193 \let\do@glo@storeentry\@gobble
```

Defined count register associated with the glossary.

```
194 \def\@gls@defsortcount##1{%
```

```
195 \expandafter\global
```

```
196 \expandafter\newcount\csname glossary@##1@sortcount\endcsname
```

```
197 }%
```

Initialise the sort key to empty.

```
198 \def\@gls@defsort##1##2{%
```

```
199 \expandafter\gdef\csname glo@##2@sort\endcsname{%
```

```
200 }%
```

If the sort key hasn't been set, increment the counter associated with the glossary and set the sort key.

```
201 \def\@gls@setsort##1{%
```

Get the parent, if one exists

```
202 \edef\@glo@parent{\csname glo@##1@parent\endcsname}%
```

Set the information for the parent entry if not already done.

```
203 \ifx\@glo@parent\@empty
```

```
204 \else
```

```
205 \expandafter\@gls@setsort\expandafter{\@glo@parent}%
```

```
206 \fi
```

Set index information for this entry

```
207 \edef\@glo@type{\csname glo@##1@type\endcsname}%
```

```
208 \edef\@gls@tmp{\csname glo@##1@sort\endcsname}%
```

```
209 \ifx\@gls@tmp\@empty
```

```
210 \expandafter\global\expandafter
```

```
211 \advance\csname glossary@\@glo@type @sortcount\endcsname by 1\relax
```

```
212 \expandafter\protected@xdef\csname glo@##1@sort\endcsname{%
```

```
213 \expandafter\glssortnumberfmt
```

```
214 {\csname glossary@\@glo@type @sortcount\endcsname}}%
```

```
215 \@glo@storeentry{##1}%
```

```
216 \fi
```

```
217 }%
```

```
218 }
```

`\glsdefmain` Define the main glossary. This will be the first glossary to be displayed when using `\printglossaries`. The default extensions conflict if used with `doc`, so provide different extensions if `doc` loaded. (If these extensions are inappropriate, use `nomain` and manually define the main glossary with the desired extensions.)

```
219 \newcommand*{\glsdefmain}{%
```

```
220 \if@gls@docloaded
```

```
221 \newglossary[glg2]{main}{gls2}{glo2}{\glossaryname}%
```

```
222 \else
```

```
223 \newglossary{main}{gls}{glo}{\glossaryname}%
```

```
224 \fi
```

Define hook to set the toc title when translator is in use.

```
225 \newcommand*{\gls@tr@set@main@toctitle}{%
```

```
226 \translatelet{\glossarytoctitle}{Glossary}%
```

```
227 }%
```

```
228 }
```

Keep track of the default glossary. This is initialised to the main glossary, but can be changed if for some reason you want to make a secondary glossary the main glossary. This affects any commands that can optionally take a glossary name as an argument (or as the value of the `type` key in a key-value list). This was mainly done so that `\loadglsentries` can temporarily change `\glsdefaulttype` while it loads a file containing new glossary entries (see [subsection 1.9](#)).

`\glsdefaulttype`

```
229 \newcommand*{\glsdefaulttype}{main}
```

Keep track of which glossary the acronyms are in. This is initialised to `\glsdefaulttype`, but is changed by the acronym package option.

`\acronymtype`

```
230 \newcommand*{\acronymtype}{\glsdefaulttype}
```

`nomain` The `nomain` option suppress the creation of the main glossary.

```
231 \@gls@declareoption{nomain}{%
```

```
232   \let\glsdefaulttype\relax
```

```
233   \renewcommand*{\glsdefmain}{}%
```

```
234 }
```

`acronym` The `acronym` option sets an associated conditional which is used in [subsection 1.16](#) to determine whether or not to define a separate glossary for acronyms.

```
235 \define@boolkey{glossaries.sty}[gls]{acronym}[true]{%
```

```
236   \ifglsacronym
```

```
237     \renewcommand*\@gls@do@acronymsdef{%
```

```
238       \DeclareAcronymList{acronym}%
```

```
239       \newglossary[alg]{acronym}{acr}{acn}{\acronymname}%
```

```
240       \renewcommand*{\acronymtype}{acronym}%
```

Define hook to set the toc title when translator is in use.

```
241       \newcommand*{\gls@tr@set@acronym@toctitle}{%
```

```
242         \translatelet{\glossarytoctitle}{Acronyms}%
```

```
243       }%
```

```
244     }%
```

```
245   \else
```

```
246     \let\@gls@do@acronymsdef\relax
```

```
247   \fi
```

```
248 }
```

`\printacronyms` Define `\printacronyms` at the start of the document if `acronym` is set and compatibility mode isn't on and `\printacronyms` hasn't already been defined.

```
249 \AtBeginDocument{%
```

```
250   \ifglsacronym
```

```
251     \ifbool{glscompatible-3.07}{%
```

```
252       }{%
```

```
253     }%
```

```
254     \providecommand*\printacronyms[1][1]{%
```

```
255       \printglossary[type=\acronymtype,#1]}%
```

```
256     }%
```

```
257   \fi
```

```
258 }
```

`@gls@do@acronymsdef` Set default value

```
259 \newcommand*{\@gls@do@acronymsdef}{}
```

acronyms Provide a synonym for acronym=true that can be passed via the document class options.

```
260 \@gls@declareoption{acronyms}{%
261   \glsacronymtrue
262   \renewcommand{\@gls@do@acronymsdef}{%
263     \DeclareAcronymList{acronym}%
264     \newglossary[alg]{acronym}{acr}{acn}{\acronymname}%
265     \renewcommand*{\acronymtype}{acronym}%
```

Define hook to set the toc title when translator is in use.

```
266     \newcommand*{\gls@tr@set@acronym@toctitle}{%
267       \translatelet{\glossarytoctitle}{Acronyms}%
268     }%
269   }%
270 }
```

\@glsacronymlists Comma-separated list of glossary labels indicating which glossaries contain acronyms. Note that \SetAcronymStyle must be used after adding labels to this macro.

```
271 \newcommand*{\@glsacronymlists}{}
```

\@addtoacronymlists

```
272 \newcommand*{\@addtoacronymlists}[1]{%
273   \ifx\@glsacronymlists\@empty
274     \protected@xdef\@glsacronymlists{#1}%
275   \else
276     \protected@xdef\@glsacronymlists{\@glsacronymlists,#1}%
277   \fi
278 }
```

\DeclareAcronymList Identifies the named glossary as a list of acronyms and adds to the list. (Doesn't check if the glossary exists, but checks if label already in list. Use \SetAcronymStyle after identifying all the acronym lists.)

```
279 \newcommand*{\DeclareAcronymList}[1]{%
280   \glsIfListOfAcronyms{#1}{\@addtoacronymlists{#1}}%
281 }
```

\glsIfListOfAcronyms

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

Determines if the glossary with the given label has been identified as being a list of acronyms.

```
282 \newcommand{\glsIfListOfAcronyms}[1]{%
283   \edef\@do@gls@islistofacronyms{%
284     \noexpand\@gls@islistofacronyms{#1}{\@glsacronymlists}}%
285   \@do@gls@islistofacronyms
286 }
```

Internal command requires label and list to be expanded:

```
287 \newcommand{\@gls@islistofacronyms}[4]{%
288   \def\gls@islistofacronyms##1,#1,##2\end@gls@islistofacronyms{%
289     \def\@before{##1}\def\@after{##2}}%
290   \gls@islistofacronyms,#2,#1,\@nil\end@gls@islistofacronyms
291   \ifx\@after\@nnil
```

Not found

```
292     #4%
293   \else
```

Found

```
294     #3%
295   \fi
296 }
```

`\if@gls@isacronymlist` Convenient boolean.

```
297 \newif\if@gls@isacronymlist
```

`\@checkisacronymlist` Sets the above boolean if argument is a label representing a list of acronyms.

```
298 \newcommand*{\@gls@checkisacronymlist}[1]{%
299   \glsIfListOfAcronyms{#1}%
300   {\@gls@isacronymlisttrue}{\@gls@isacronymlistfalse}%
301 }
```

`\SetAcronymLists` Sets the “list of acronyms” list. Argument must be a comma-separated list of glossary labels. (Doesn’t check at this point if the glossaries exists.)

```
302 \newcommand*{\SetAcronymLists}[1]{%
303   \renewcommand*{\@gls@acronymlists}{#1}%
304 }
```

`acronymlists`

```
305 \define@key{glossaries.sty}{acronymlists}{%
306   \DeclareAcronymList{#1}%
307 }
```

The default counter associated with the numbers in the glossary is stored in `\glscounter`. This is initialised to the page counter. This is used as the default counter when a new glossary is defined, unless a different counter is specified in the optional argument to `\newglossary` (see [subsection 1.6](#)).

`\glscounter`

```
308 \newcommand{\glscounter}{page}
```

`counter` The counter option changes the default counter. (This just redefines `\glscounter`.)

```
309 \define@key{glossaries.sty}{counter}{%
310   \renewcommand*{\glscounter}{#1}%
311 }
```

```

\@gls@nohyperlist
312 \newcommand*{\@gls@nohyperlist}{}

sDeclareNoHyperList
313 \newcommand*{\GlsDeclareNoHyperList}[1]{%
314   \ifdefempty\@gls@nohyperlist
315   {%
316     \renewcommand*{\@gls@nohyperlist}{#1}%
317   }%
318   {%
319     \appto\@gls@nohyperlist{,#1}%
320   }%
321 }

nohypertypes
322 \define@key{glossaries.sty}{nohypertypes}{%
323   \GlsDeclareNoHyperList{#1}%
324 }

\GlossariesWarning Prints a warning message.
325 \newcommand*{\GlossariesWarning}[1]{%
326   \PackageWarning{glossaries}{#1}%
327 }

sariesWarningNoLine Prints a warning message without the line number.
328 \newcommand*{\GlossariesWarningNoLine}[1]{%
329   \PackageWarningNoLine{glossaries}{#1}%
330 }

nowarn Define package option to suppress warnings
331 \@gls@declareoption{nowarn}{%
332   \renewcommand*{\GlossariesWarning}[1]{}%
333   \renewcommand*{\GlossariesWarningNoLine}[1]{}%
334 }

@warnonglossdefined Issue a warning if overriding \printglossary
335 \newcommand*{\@gls@warnonglossdefined}{%
336   \GlossariesWarning{Overriding \string\printglossary}%
337 }

rnontheglossdefined Issue a warning if overriding theglossary
338 \newcommand*{\@gls@warnontheGLOSSdefined}{%
339   \GlossariesWarning{Overriding 'theglossary' environment}%
340 }

noredefwarn Suppress warning on redefinition of \printglossary
341 \@gls@declareoption{noredefwarn}{%
342   \renewcommand*{\@gls@warnonglossdefined}{}%

```



```

343 \renewcommand*{\@gls@warnontheglossdefined}{}%
344 }

```

As from version 3.08a, the only information written to the external glossary files are the label and sort values. Therefore, now, the only sanitize option that makes sense is the one for the sort key. so the sanitize option is now deprecated and there is only a sanitizesort option.

\@gls@sanitizedesc

```

345 \newcommand*{\@gls@sanitizedesc}{%
346 }

```

\glssetexpandfield

```
\glssetexpandfield{<field>}
```

Sets field to always expand.

```

347 \newcommand*{\glssetexpandfield}[1]{%
348   \csdef{gls@assign@#1@field}##1##2{%
349     \@gls@expand@field{##1}{#1}{##2}%
350   }%
351 }

```

\glssetnoexpandfield

```
\glssetnoexpandfield{<field>}
```

Sets field to never expand.

```

352 \newcommand*{\glssetnoexpandfield}[1]{%
353   \csdef{gls@assign@#1@field}##1##2{%
354     \@gls@noexpand@field{##1}{#1}{##2}%
355   }%
356 }

```

s@assign@type@field The type must always be expandable.

```
357 \glssetexpandfield{type}
```

s@assign@desc@field The description is not expanded by default:

```
358 \glssetnoexpandfield{desc}
```

gn@descplural@field

```
359 \glssetnoexpandfield{descplural}
```

\@gls@sanitizename

```
360 \newcommand*{\@gls@sanitizename}{}
```

s@assign@name@field Don't expand name by default.

```
361 \glssetnoexpandfield{name}
```

@gls@sanitizesymbol

```
362 \newcommand*{\@gls@sanitizesymbol}{}
```

assign@symbol@field Don't expand symbol by default.

```
363 \glssetnoexpandfield{symbol}
```

@symbolplural@field

```
364 \glssetnoexpandfield{symbolplural}
```

Sanitizing stuff:

\@gls@sanitizesort

```
365 \newcommand*{\@gls@sanitizesort}{%
366   \ifglssanitizesort
367     \@gls@sanitizesort
368   \else
369     \@gls@nosanitizesort
370   \fi
371 }
```

\@@gls@sanitizesort

```
372 \newcommand*\@@gls@sanitizesort{%
373   \@onelevel@sanitize\@glo@sort
374 }
```

@gls@nosanitizesort

```
375 \newcommand*{\@gls@nosanitizesort}{}
```

@noidx@sanitizesort Remove braces around first character (if present) before sanitizing.

```
376 \newcommand*\@gls@noidx@sanitizesort{%
377   \ifdefvoid\@glo@sort
378   }%
379   {%
380     \expandafter\@gls@noidx@sanitizesort\@glo@sort\gls@end@sanitizesort
381   }%
382 }
383 \def\@@gls@noidx@sanitizesort#1#2\gls@end@sanitizesort{%
384   \def\@glo@sort{#1#2}%
385   \@onelevel@sanitize\@glo@sort
386 }
```

@noidx@nosanitizesort

```
387 \newcommand*\@gls@noidx@nosanitizesort{%
388   \ifdefvoid\@glo@sort
389   }%
390   {%
391     \expandafter\@gls@noidx@no@sanitizesort\@glo@sort\gls@end@sanitizesort
392   }%
```

```

393 }
394 \def\@@gls@noidx@no@sanitizesort#1#2\gls@end@sanitizesort{%
395   \bgroup
396     \glsnoidxstripaccents
397     \protected@xdef\@@glo@sort{#1#2}%
398   \egroup
399   \let\@glo@sort\@@glo@sort
400 }

```

lsglsnoidxstripaccents

```

401 \newcommand*\glsnoidxstripaccents{%
402   \let\IeC\@firstofone
403   \let\''\@firstofone
404   \let\'\@firstofone
405   \let\~\@firstofone
406   \let\""\@firstofone
407   \let\u\@firstofone
408   \let\t\@firstofone
409   \let\d\@firstofone
410   \let\r\@firstofone
411   \let\=\@firstofone
412   \let\.\@firstofone
413   \let\~\@firstofone
414   \let\v\@firstofone
415   \let\H\@firstofone
416   \let\c\@firstofone
417   \let\b\@firstofone
418   \def\AE{AE}%
419   \def\ae{ae}%
420   \def\OE{OE}%
421   \def\oe{oe}%
422   \def\AA{AA}%
423   \def\aa{aa}%
424   \def\L{L}%
425   \def\l{l}%
426   \def\O{O}%
427   \def\o{o}%
428   \def\SS{SS}%
429   \def\ss{ss}%
430   \def\th{th}%
431 }

```

Before defining the sanitize package option, The key-value list for the sanitize value needs to be defined. These are all boolean keys. If they are not given a value, assume true.

```

432 \define@boolkey[gls]{sanitize}{description}[true]{%
433   \GlossariesWarning{sanitize={description} package option deprecated}%
434   \ifgls@sanitize@description
435     \glssetnoexpandfield{desc}%

```

```

436   \glssetnoexpandfield{descplural}%
437 \else
438   \glssetexpandfield{desc}%
439   \glssetexpandfield{descplural}%
440 \fi
441 }

442 \define@boolkey[glS]{sanitize}{name}[true]{%
443   \GlossariesWarning{sanitize={name} package option deprecated}%
444   \ifglS@sanitize@name
445     \glssetnoexpandfield{name}%
446   \else
447     \glssetexpandfield{name}%
448   \fi
449 }

450 \define@boolkey[glS]{sanitize}{symbol}[true]{%
451   \GlossariesWarning{sanitize={symbol} package option deprecated}%
452   \ifglS@sanitize@symbol
453     \glssetnoexpandfield{symbol}%
454     \glssetnoexpandfield{symbolplural}%
455   \else
456     \glssetexpandfield{symbol}%
457     \glssetexpandfield{symbolplural}%
458   \fi
459 }

```

sanitizesort

```

460 \define@boolkey{glossaries.sty}[glS]{sanitizesort}[true]{%
461   \ifglssanitizesort
462     \glssetnoexpandfield{sortvalue}%
463     \renewcommand*{\@glS@noidx@setsanitizesort}{%
464       \glssanitizesorttrue
465       \glssetnoexpandfield{sortvalue}%
466     }%
467   \else
468     \glssetexpandfield{sortvalue}%
469     \renewcommand*{\@glS@noidx@setsanitizesort}{%
470       \glssanitizesortfalse
471       \glssetexpandfield{sortvalue}%
472     }%
473   \fi
474 }

```

Default setting:

```

475 \glssanitizesorttrue
476 \glssetnoexpandfield{sortvalue}%

```

`\@glS@setsanitizesort` Default behaviour for `\makenoidxglossaries` is `sanitizesort=false`.

```

477 \newcommand*{\@glS@noidx@setsanitizesort}{%
478   \glssanitizesortfalse

```

```

479 \glssetexpandfield{sortvalue}%
480 }

481 \define@choicekey[gls]{sanitize}{sort}{true,false}[true]{%
482 \setbool{glssanitizesort}{#1}%
483 \ifglssanitizesort
484 \glssetnoexpandfield{sortvalue}%
485 \else
486 \glssetexpandfield{sortvalue}%
487 \fi
488 \GlossariesWarning{sanitize={sort} package option
489 deprecated. Use sanitizesort instead}%
490 }

```

sanitize

```

491 \define@key{glossaries.sty}{sanitize}[description=true,symbol=true,
492 name=true]{%
493 \ifthenelse{\equal{#1}{none}}{%
494 {%
495 \GlossariesWarning{sanitize package option deprecated}%
496 }%
497 {%
498 \setkeys[gls]{sanitize}{#1}%
499 }%
500 }

```

`\ifglstranslate` As from version 3.13a, the translator package option is a choice rather than boolean option so now need to define conditional:

```
501 \newif\ifglstranslate
```

`ls@nottranslatorhook`

```
502 \newcommand*{\@gls@nottranslatorhook{}
```

`nottranslate` Provide a synonym for `translate=false` that can be passed via the document class.

```

503 \@gls@declareoption{nottranslate}{%
504 \glstranslatefalse
505 \let\@gls@nottranslatorhook\relax
506 }

```

`translate` Define translate option. If false don't set up multi-lingual support.

```

507 \define@choicekey{glossaries.sty}{translate}[\val\nr]{%
508 {true,false,babel}[true]%
509 {%
510 \ifcase\nr\relax
511 \glstranslatetrue
512 \or
513 \glstranslatefalse
514 \let\@gls@nottranslatorhook\relax

```

```

515 \or
516 \glstranslatefalse
517 \def\@gls@notranslatorhook{\RequirePackage{glossaries-babel}}%
518 \fi
519 }

```

Set the default value:

```

520 \glstranslatefalse
521 \@ifpackageloaded{translator}%
522 {\glstranslatetrue}%
523 {%
524 \ifpackageloaded{polyglossia}%
525 {\glstranslatetrue}%
526 {%
527 \ifpackageloaded{babel}{\glstranslatetrue}{}}%
528 }%
529 }

```

`indexonlyfirst` Set whether to only index on first use.

```

530 \define@boolkey{glossaries.sty}[gls]{indexonlyfirst}[true]{}
531 \glsindexonlyfirstfalse

```

`hyperfirst` Set whether or not terms should have a hyperlink on first use.

```

532 \define@boolkey{glossaries.sty}[gls]{hyperfirst}[true]{}
533 \glshyperfirsttrue

```

`\@gls@setacrstyle` Keep track of whether an acronym style has been set (for the benefit of `\setupglossaries`):

```

534 \newcommand*\@gls@setacrstyle{}

```

`footnote` Set the long form of the acronym in footnote on first use.

```

535 \define@boolkey{glossaries.sty}[glsacr]{footnote}[true]{%
536 \ifbool{glsacrdescription}%
537 {}%
538 {%
539 \renewcommand*\@gls@sanitizedesc}{}%
540 }%
541 \renewcommand*\@gls@setacrstyle{\SetAcronymStyle}%
542 }

```

`description` Allow acronyms to have a description (needs to be set using the description key in the optional argument of `\newacronym`).

```

543 \define@boolkey{glossaries.sty}[glsacr]{description}[true]{%
544 \renewcommand*\@gls@sanitizesymbol}{}%
545 \renewcommand*\@gls@setacrstyle{\SetAcronymStyle}%
546 }

```

smallcaps Define `\newacronym` to set the short form in small capitals.

```

547 \define@boolkey{glossaries.sty}[glsacr]{smallcaps}[true]{%
548   \renewcommand*{\@gls@sanitizesymbol}{}%
549   \renewcommand*{\@gls@setacrstyle}{\SetAcronymStyle}%
550 }

```

smaller Define `\newacronym` to set the short form using `\smaller` which obviously needs to be defined by loading the appropriate package.

```

551 \define@boolkey{glossaries.sty}[glsacr]{smaller}[true]{%
552   \renewcommand*{\@gls@sanitizesymbol}{}%
553   \renewcommand*{\@gls@setacrstyle}{\SetAcronymStyle}%
554 }

```

dua Define `\newacronym` to always use the long forms (i.e. don't use acronyms)

```

555 \define@boolkey{glossaries.sty}[glsacr]{dua}[true]{%
556   \renewcommand*{\@gls@sanitizesymbol}{}%
557   \renewcommand*{\@gls@setacrstyle}{\SetAcronymStyle}%
558 }

```

shortcuts Define acronym shortcuts.

```

559 \define@boolkey{glossaries.sty}[glsacr]{shortcuts}[true]{}

```

\glsorder Stores the glossary ordering. This may either be “word” or “letter”. This passes the relevant information to `makeglossaries`. The default is word ordering.

```

560 \newcommand*{\glsorder}{word}

```

\@glsorder The ordering information is written to the auxiliary file for `makeglossaries`, so ignore the auxiliary information.

```

561 \newcommand*{\@glsorder}[1]{}

```

order

```

562 \define@choicekey{glossaries.sty}{order}{word,letter}{%
563   \def\glsorder{#1}}

```

\ifglxindy Provide boolean to determine whether `xindy` or `makeindex` will be used to sort the glossaries.

```

564 \newif\ifglxindy

```

The default is `makeindex`:

```

565 \glxindyfalse

```

makeindex Define package option to specify that `makeindex` will be used to sort the glossaries:

```

566 \@gls@declareoption{makeindex}{\glxindyfalse}

```

The xindy package option may have a value which in turn can be a key=value list. First define the keys for this sub-list. The boolean glsnumbers determines whether to automatically add the glsnumbers letter group.

```
567 \define@boolkey[gls]{xindy}{glsnumbers}[true]{}
568 \gls{xindy@glsnumberstrue}
```

`\@xdy@main@language` Define what language to use for each glossary type (if a language is not defined for a particular glossary type the language specified for the main glossary is used.)

```
569 \def\@xdy@main@language{\language}%
```

Define key to set the language

```
570 \define@key[gls]{xindy}{language}{\def\@xdy@main@language{#1}}
```

`\gls@codepage` Define the code page. If `\inputencodingname` is defined use that, otherwise have initialise with no codepage.

```
571 \ifcsundef{inputencodingname}{%
572   \def\gls@codepage{}}{%
573   \def\gls@codepage{\inputencodingname}
574 }
```

Define a key to set the code page.

```
575 \define@key[gls]{xindy}{codepage}{\def\gls@codepage{#1}}
```

`xindy` Define package option to specify that xindy will be used to sort the glossaries:

```
576 \define@key{glossaries.sty}{xindy}[]{%
577   \glsxindytrue
578   \setkeys[gls]{xindy}{#1}%
579 }
```

`xindygloss` Provide a synonym for xindy that can be passed via the document class options.

```
580 \@gls@declareoption{xindygloss}{%
581   \glsxindytrue
582 }
```

`xindynoglsnumbers` Provide a synonym for `xindy=glsnumbers=false` that can be passed via the document class options.

```
583 \@gls@declareoption{xindynoglsnumbers}{%
584   \glsxindytrue
585   \gls{xindy@glsnumbersfalse}
586 }
```

`automake` If this setting is on, automatically run **makeindex/xindy** at the end of the document. Must be used with `\makeglossaries`. Default is false.

```
587 \define@boolkey{glossaries.sty}[gls]{automake}[true]{%
588   \ifglautomake
589     \renewcommand*{\@gls@doautomake}{%}
```



```

590     \PackageError{glossaries}{You must use
591     \string\makeglossaries\space with automake=true}
592     {%
593         Either remove the automake=true setting or
594         add \string\makeglossaries\space to your document preamble.%
595     }%
596 }%
597 \else
598     \renewcommand*{\@gls@doautomake}{}%
599 \fi
600 }
601 \glsautomakefalse

```

\@gls@doautomake

```

602 \newcommand*{\@gls@doautomake}{}
603 \AtEndDocument{\@gls@doautomake}

```

savewrites The savewrites package option is provided to save on the number of write registers.

```

604 \define@boolkey{glossaries.sty}[gls]{savewrites}[true]{%
605     \ifglssavewrites
606         \renewcommand*{\glswritefiles}{\@glswritefiles}%
607     \else
608         \let\glswritefiles\@empty
609     \fi
610 }

```

Set default:

```

611 \glssavewritesfalse
612 \let\glswritefiles\@empty

```

compatible-3.07

```

613 \define@boolkey{glossaries.sty}[gls]{compatible-3.07}[true]{%
614 \boolfalse{glscompatible-3.07}

```

compatible-2.07

```

615 \define@boolkey{glossaries.sty}[gls]{compatible-2.07}[true]{%
    Also set 3.07 compatibility if this option is set.
616     \ifbool{glscompatible-2.07}%
617     {%
618         \booltrue{glscompatible-3.07}%
619     }%
620     {%
621 }
622 \boolfalse{glscompatible-2.07}

```

symbols Create a “symbols” glossary type

```

623 \@gls@declareoption{symbols}{%

```

```

624 \let\@gls@do@symbolsdef\@gls@symbolsdef
625 }

```

Default is not to define the symbols glossary:

```

626 \newcommand*\@gls@do@symbolsdef{}\}

```

\@gls@symbolsdef

```

627 \newcommand*\@gls@symbolsdef{%
628   \newglossary[slg]{symbols}{sls}{slo}{\glssymbolsgroupname}%
629   \newcommand*\@printsymbols[1][\printglossary[type=symbols,##1]]%

  Define hook to set the toc title when translator is in use.
630   \newcommand*\@gls@tr@set@symbols@toctitle{%
631     \translatelet{\glossarytoctitle}{Symbols (glossaries)}%
632   }%
633 }%

```

numbers Create a “symbols” glossary type

```

634 \@gls@declareoption{numbers}{%
635   \let\@gls@do@numbersdef\@gls@numbersdef
636 }

```

Default is not to define the numbers glossary:

```

637 \newcommand*\@gls@do@numbersdef{}\}

```

\@gls@numbersdef

```

638 \newcommand*\@gls@numbersdef{%
639   \newglossary[nlg]{numbers}{nls}{nlo}{\glsnumbersgroupname}%
640   \newcommand*\@printnumbers[1][\printglossary[type=numbers,##1]]%

  Define hook to set the toc title when translator is in use.
641   \newcommand*\@gls@tr@set@numbers@toctitle{%
642     \translatelet{\glossarytoctitle}{Numbers (glossaries)}%
643   }%
644 }%

```

index Create an “index” glossary type

```

645 \@gls@declareoption{index}{%
646   \let\@gls@do@indexdef\@gls@indexdef
647 }

```

Default is not to define index glossary:

```

648 \newcommand*\@gls@do@indexdef{}\}

```

\@gls@indexdef \indexname isn't set by glossaries.

```

649 \newcommand*\@gls@indexdef{%
650   \newglossary[ilg]{index}{ind}{idx}{\indexname}%
651   \newcommand*\@printindex[1][\printglossary[type=index,##1]]%
652   \newcommand*\@newterm[2][\%
653     \newglossaryentry{##2}%
654     {type={index},name={##2},description={\nopostdesc},##1}}
655 }%

```

Process package options. First process any options that have been passed via the document class.

```

656 \@for\CurrentOption :=\@declaredoptions\do{%
657   \ifx\CurrentOption\@empty
658   \else
659     \@expandtwoargs
660     \in@ {,\CurrentOption ,}{,\@classoptionslist,\@curroptions,}%
661     \ifin@
662     \@use@option
663     \expandafter \let\csname ds@\CurrentOption\endcsname\@empty
664     \fi
665   \fi
666 }

```

Now process options passed to the package:

```
667 \ProcessOptionsX
```

Load backward compatibility stuff:

```
668 \RequirePackage{glossaries-compatible-307}
```

`\setupglossaries` Provide way to set options after package has been loaded. However, some options must be set before `\ProcessOptionsX`, so they have to be disabled:

```

669 \disable@keys{glossaries.sty}{compatible-2.07,%
670 xindy,xindygloss,xindynoglsnumbers,makeindex,%
671 acronym,translate,nottranslate,nolong,nosuper,notree,nostyles,nomain}

```

Now define `\setupglossaries`:

```

672 \newcommand*\setupglossaries}[1]{%
673   \renewcommand*\@gls@setacrstyle}{}%
674   \ifglsacrshortcuts
675     \def\@gls@setupshortcuts{\glsacrshortcutstrue}%
676   \else
677     \def\@gls@setupshortcuts{%
678       \ifglsacrshortcuts
679         \DefineAcronymSynonyms
680       \fi
681     }%
682   \fi
683   \glsacrshortcutsfalse
684   \let\@gls@do@numbersdef\relax
685   \let\@gls@do@symbolssdef\relax
686   \let\@gls@do@indexdef\relax
687   \let\@gls@do@acronymsdef\relax
688   \setkeys{glossaries.sty}{#1}%
689   \@gls@setacrstyle
690   \@gls@setupshortcuts
691   \@gls@do@acronymsdef
692   \@gls@do@numbersdef
693   \@gls@do@symbolssdef
694   \@gls@do@indexdef

```

695 }

If package is loaded, check to see if is installed, but only if translation is required.

```

696 \ifglstranslate
697   \@ifpackageloaded{polyglossia}%
698   {%
    polyglossia fakes babel so need to check for polyglossia first.
699   }%
700   {%
701     \@ifpackageloaded{babel}%
702     {%
703       \IfFileExists{translator.sty}%
704       {%
705         \RequirePackage{translator}%
706       }%
707     }%
708   }%
709   {}
710 }
711 \fi

```

If chapters are defined and the user has requested the section counter as a package option, `\@chapter` will be modified so that it adds a `section.<n>.0` target, otherwise entries placed before the first section of a chapter will have undefined links.

The same problem will also occur if a lower sectional unit is used, but this is less likely to happen. If it does, or if you change `\glscounter` to `section` later, you will have to specify a different counter for the entries that give rise to a name{<section-level>.<n>.0} non-existent warning (e.g. `\gls[counter=chapter]{label}`).

```

712 \ifthenelse{\equal{\glscounter}{section}}{%
713   {%
714     \ifcsundef{chapter}{}%
715     {%
716       \let\@gls@old@chapter\@chapter
717       \def\@chapter[#1]#2{\@gls@old@chapter[#1]{#2}%
718         \ifcsundef{hyperdef}{}{\hyperdef{section}{\thesection}}}%
719     }%
720   }%
721   {}

```

`\@gls@onlypremakeg` Some commands only have an effect when used before `\makeglossaries`. So define a list of commands that should be disabled after `\makeglossaries`

```

722 \newcommand*{\@gls@onlypremakeg}{}

```

`\@onlypremakeg` Adds the specified control sequence to the list of commands that must be disabled after `\makeglossaries`.

```

723 \newcommand*{\@onlypremakeg}[1]{%
724   \ifx\@gls@onlypremakeg\@empty
725     \def\@gls@onlypremakeg{#1}%
726   \else
727     \expandafter\toks@\expandafter{\@gls@onlypremakeg}%
728     \edef\@gls@onlypremakeg{\the\toks@,\noexpand#1}%
729   \fi
730 }

```

`\@disable@onlypremakeg` Disable all commands listed in `\@gls@onlypremakeg`

```

731 \newcommand*{\@disable@onlypremakeg}{%
732 \@for\@thiscs:=\@gls@onlypremakeg\do{%
733   \expandafter\@disable@premakecs\@thiscs%
734 }}

```

`\@disable@premakecs` Disables the given command.

```

735 \newcommand*{\@disable@premakecs}[1]{%
736   \def#1{\PackageError{glossaries}{\string#1\space may only be
737     used before \string\makeglossaries}{You can't use
738     \string#1\space after \string\makeglossaries}}%
739 }

```

1.3 Default values

This section sets up default values that are used by this package. Some of the names may already be defined (e.g. by) so `\providecommand` is used.

Main glossary title:

`\glossaryname`

```

740 \providecommand*{\glossaryname}{Glossary}

```

The title for the acronym glossary type (which is defined if acronym package option is used) is given by `\acronymname`. If the acronym package option is not used, `\acronymname` won't be used.

`\acronymname`

```

741 \providecommand*{\acronymname}{Acronyms}

```

`\glssettoctitle` Sets the TOC title for the given glossary.

```

742 \newcommand*{\glssettoctitle}[1]{%
743   \def\glossarytoctitle{\curname @glotype@#1@title\endcurname}}

```

The following commands provide text for the headers used by some of the tabular-like glossary styles. Whether or not they get used in the glossary depends on the glossary style.

`\entryname`

```

744 \providecommand*{\entryname}{Notation}

```

`\descriptionname`

```
745 \providecommand*{\descriptionname}{Description}
```

`\symbolname`

```
746 \providecommand*{\symbolname}{Symbol}
```

`\pagelistname`

```
747 \providecommand*{\pagelistname}{Page List}
```

Labels for `makeindex`'s symbol and number groups:

`glssymbolsgroupname`

```
748 \providecommand*{\glssymbolsgroupname}{Symbols}
```

`glsnumbersgroupname`

```
749 \providecommand*{\glsnumbersgroupname}{Numbers}
```

`\glspluralsuffix` The default plural is formed by appending `\glspluralsuffix` to the singular form.

```
750 \newcommand*{\glspluralsuffix}{s}
```

`\seename`

```
751 \providecommand*{\seename}{see}
```

`\andname`

```
752 \providecommand*{\andname}{\&}
```

Add multi-lingual support. Thanks to everyone who contributed to the translations from both `comp.text.tex` and via email.

`dglossarytocaptions` If using , `\glossaryname` should be defined in terms of `\translate`, but if `babel` is also loaded, it will redefine `\glossaryname` whenever the language is set, so override it. (Don't use `\addto` as doesn't define it.)

```
753 \newcommand*{\addglossarytocaptions}[1]{%
754   \ifcsundef{captions#1}{}%
755   {%
756     \expandafter\let\expandafter\@gls@tmp\csname captions#1\endcsname
757     \expandafter\toks@\expandafter{\@gls@tmp
758       \renewcommand*{\glossaryname}{\translate{Glossary}}}%
759     }%
760     \expandafter\edef\csname captions#1\endcsname{\the\toks@}%
761   }%
762 }

763 \ifglstranslate
```

If is not install, used standard captions, otherwise load dictionary.

```

764 \@ifpackageloaded{translator}{%
765   \usedictionary{glossaries-dictionary}%
766   \addglossarytocaptions{portuges}%
767   \addglossarytocaptions{portuguese}%
768   \addglossarytocaptions{brazil}%
769   \addglossarytocaptions{brazilian}%
770   \addglossarytocaptions{danish}%
771   \addglossarytocaptions{dutch}%
772   \addglossarytocaptions{afrikaans}%
773   \addglossarytocaptions{english}%
774   \addglossarytocaptions{UKenglish}%
775   \addglossarytocaptions{USenglish}%
776   \addglossarytocaptions{american}%
777   \addglossarytocaptions{australian}%
778   \addglossarytocaptions{british}%
779   \addglossarytocaptions{canadian}%
780   \addglossarytocaptions{newzealand}%
781   \addglossarytocaptions{french}%
782   \addglossarytocaptions{frenchb}%
783   \addglossarytocaptions{francais}%
784   \addglossarytocaptions{acadian}%
785   \addglossarytocaptions{canadien}%
786   \addglossarytocaptions{german}%
787   \addglossarytocaptions{germanb}%
788   \addglossarytocaptions{austrian}%
789   \addglossarytocaptions{naustrian}%
790   \addglossarytocaptions{ngerman}%
791   \addglossarytocaptions{irish}%
792   \addglossarytocaptions{italian}%
793   \addglossarytocaptions{magyar}%
794   \addglossarytocaptions{hungarian}%
795   \addglossarytocaptions{polish}%
796   \addglossarytocaptions{spanish}%
797   \renewcommand*{\glssettoctitle}[1]{%
798     \ifcsdef{gls@tr@set@#1@toctitle}%
799     {%
800       \csuse{gls@tr@set@#1@toctitle}%
801     }%
802     {%
803       \def\glossarytoctitle{\csname @glotype@#1@title\endcsname}%
804     }%
805   }%
806   \renewcommand*{\glossaryname}{\translate{Glossary}}%
807   \renewcommand*{\acronymname}{\translate{Acronyms}}%
808   \renewcommand*{\entryname}{\translate{Notation (glossaries)}}%
809   \renewcommand*{\descriptionname}{%
810     \translate{Description (glossaries)}}%
811   \renewcommand*{\symbolname}{\translate{Symbol (glossaries)}}%

```

```

812 \renewcommand*{\pagelistname}{%
813 \translate{Page List (glossaries)}}%
814 \renewcommand*{\glssymbolsgroupname}{%
815 \translate{Symbols (glossaries)}}%
816 \renewcommand*{\glsnumbersgroupname}{%
817 \translate{Numbers (glossaries)}}%
818 }{%

819 \@ifpackageloaded{polyglossia}%
820 {\RequirePackage{glossaries-polyglossia}}%
821 {%
822 \@ifpackageloaded{babel}{%
823 \RequirePackage{glossaries-babel}}}%
824 }}
825 \else

826 \@gls@notranslatorhook
827 \fi

```

`\nopostdesc` Provide a means to suppress description terminator for a given entry. (Useful for entries with no description.) Has no effect outside the glossaries.

```
828 \DeclareRobustCommand*{\nopostdesc}{}
```

`\@nopostdesc` Suppress next description terminator.

```

829 \newcommand*{\@nopostdesc}{%
830 \let\org@glspostdescription\glspostdescription
831 \def\glspostdescription{%
832 \let\glspostdescription\org@glspostdescription}%
833 }

```

`\@no@post@desc` Used for comparison purposes.

```
834 \newcommand*{\@no@post@desc}{\nopostdesc}
```

`\glspar` Provide means of having a paragraph break in glossary entries

```
835 \newcommand{\glspar}{\par}
```

`\setStyleFile` Sets the style file. The relevant extension is appended.

```

836 \newcommand{\setStyleFile}[1]{%
837 \renewcommand*{\gls@istfilebase}{#1}%
  Just in case \istfilename has been modified.
838 \ifglsxindy
839 \def\istfilename{\gls@istfilebase.xdy}
840 \else
841 \def\istfilename{\gls@istfilebase.ist}
842 \fi
843 }

```

This command only has an effect prior to using `\makeglossaries`.

```
844 \@onlypremakeg\setStyleFile
```


The name of the makeindex or xindy style file is given by `\istfilename`. This file is created by `\writeist` (which is used by `\makeglossaries`) so redefining this command will only have an effect if it is done *before* `\makeglossaries`. As from v1.17, use `\setStyleFile` instead of directly redefining `\istfilename`.

`\istfilename`

```
845 \ifglsxindy
846   \def\istfilename{\gls@istfilebase.xdy}
847 \else
848   \def\istfilename{\gls@istfilebase.ist}
849 \fi
```

`\gls@istfilebase`

```
850 \newcommand*{\gls@istfilebase}{\jobname}
```

The `makeglossaries` Perl script picks up this name from the auxiliary file. If the name ends with `.xdy` it calls `xindy` otherwise it calls `makeindex`. Since its not required by \TeX , `\@istfilename` ignores its argument.

`\@istfilename`

```
851 \newcommand*{\@istfilename}[1]{}
```

This command is the value of the `page_compositor` `makeindex` key. Again, any redefinition of this command must take place *before* `\writeist` otherwise it will have no effect. As from 1.17, use `\glsSetCompositor` instead of directly redefining `\glscompositor`.

`\glscompositor`

```
852 \newcommand*{\glscompositor}{.}
```

`\glsSetCompositor` Sets the compositor.

```
853 \newcommand*{\glsSetCompositor}[1]{%
854   \renewcommand*{\glscompositor}{#1}}
```

Only use before `\makeglossaries`

```
855 \@onlypremakeg\glsSetCompositor
```

(The page compositor is usually defined as a dash when using `makeindex`, but most of the standard counters used by \TeX use a full stop as the compositor, which is why I have used it as the default.) If `xindy` is used `\glscompositor` only affects the `arabic-page-numbers` location class.

`@glsAlphacompositor`

This is only used by `xindy`. It specifies the compositor to use when location numbers are in the form `\langle letter \rangle \langle compositor \rangle \langle number \rangle`. For example, if `\@glsAlphacompositor` is set to `."` then it allows locations such as `A.1` whereas if `\@glsAlphacompositor` is set to `-"` then it allows locations such as `A-1`.

```
856 \newcommand*{\@glsAlphacompositor}{\glscompositor}
```

`\glsSetAlphaCompositor` Sets the alpha compositor.

```
857 \ifglxsindy
858   \newcommand*\glsSetAlphaCompositor[1]{%
859     \renewcommand*\@glsAlphacompositor{#1}}
860 \else
861   \newcommand*\glsSetAlphaCompositor[1]{%
862     \glsnoxywarning\glsSetAlphaCompositor}
863 \fi
```

Can only be used before `\makeglossaries`

```
864 \@onlypremakeg\glsSetAlphaCompositor
```

`\gls@suffixF` Suffix to use for a two page list. This overrides the separator and the closing page number if set to something other than an empty macro.

```
865 \newcommand*\{\gls@suffixF}{}
```

`\glsSetSuffixF` Sets the suffix to use for a two page list.

```
866 \newcommand*\{\glsSetSuffixF}[1]{%
867   \renewcommand*\{\gls@suffixF}{#1}}
```

Only has an effect when used before `\makeglossaries`

```
868 \@onlypremakeg\glsSetSuffixF
```

`\gls@suffixFF` Suffix to use for a three page list. This overrides the separator and the closing page number if set to something other than an empty macro.

```
869 \newcommand*\{\gls@suffixFF}{}
```

`\glsSetSuffixFF` Sets the suffix to use for a three page list.

```
870 \newcommand*\{\glsSetSuffixFF}[1]{%
871   \renewcommand*\{\gls@suffixFF}{#1}%
872 }
```

`\glsnumberformat` The command `\glsnumberformat` indicates the default format for the page numbers in the glossary. (Note that this is not the same as `\glossaryentrynumbers`, but applies to individual numbers or groups of numbers within an entry's associated number list.) If hyperlinks are defined, it will use `\glshypernumber`, otherwise it will simply display its argument “as is”.

```
873 \ifcsundef{hyperlink}%
874 {%
875   \newcommand*\{\glsnumberformat}[1]{#1}%
876 }%
877 {%
878   \newcommand*\{\glsnumberformat}[1]{\glshypernumber{#1}}%
879 }
```

Individual numbers in an entry's associated number list are delimited using `\delimN` (which corresponds to the `delim_n makeindex` keyword). The default value is a comma followed by a space.

`\delimN`

```
880 \newcommand{\delimN}{, }
```

A range of numbers within an entry's associated number list is delimited using `\delimR` (which corresponds to the `delim_r makeindex` keyword). The default is an en-dash.

`\delimR`

```
881 \newcommand{\delimR}{--}
```

The glossary preamble is given by `\glossarypreamble`. This will appear after the glossary sectioning command, and before the `theglossary` environment. It is designed to allow the user to add information pertaining to the glossary (e.g. "page numbers in italic indicate the primary definition") therefore `\glossarypreamble` shouldn't be affected by the glossary style. (So if you define your own glossary style, don't have it change `\glossarypreamble`.) The preamble is empty by default. If you have multiple glossaries, and you want a different preamble for each glossary, you will need to use `\printglossary` for each glossary type, instead of `\printglossaries`, and redefine `\glossarypreamble` before each `\printglossary`.

`\glossarypreamble`

```
882 \newcommand*{\glossarypreamble}{%
883   \csuse{@glossarypreamble@currentglossary}%
884 }
```

`\setglossarypreamble` `\setglossarypreamble[<type>]{<text>}`

Code provided by Michael Pock.

```
885 \newcommand{\setglossarypreamble}[2][\glsdefaulttype]{%
886   \ifglossaryexists{#1}{%
887     \csgdef{@glossarypreamble@#1}{#2}%
888   }{%
889     \GlossariesWarning{%
890       Glossary ‘#1’ is not defined%
891     }%
892   }%
893 }
```

The glossary postamble is given by `\glossarypostamble`. This is provided to allow the user to add something after the end of the `theglossary` environment (again, this shouldn't be affected by the glossary style). It is, of course, possible to simply add the text after `\printglossary`, but if you only want the postamble to appear after the first glossary, but not after subsequent glossaries, you can do something like:

```
\renewcommand{\glossarypostamble}{For a complete list of terms
see \cite{blah}\gdef\glossarypreamble{}}
```

```

\glossarypostamble
894 \newcommand*{\glossarypostamble}{}

\glossarysection The sectioning command that starts a glossary is given by \glossarysection.
                  (This does not form part of the glossary style, and so should not be changed by
                  a glossary style.) If \phantomsection is defined, it uses \p@glossarysection,
                  otherwise it uses \@glossarysection.
895 \newcommand*{\glossarysection}[2][\@gls@title]{%
896   \def\@gls@title{#2}%
897   \ifcsundef{phantomsection}%
898     {%
899       \@glossarysection{#1}{#2}%
900     }%
901     {%
902       \@p@glossarysection{#1}{#2}%
903     }%

904   \glsglossarymark{\glossarytoctitle}%
905 }

\glsglossarymark Sets the header mark for the glossary. Takes the glossary short (TOC) title as the
                  argument.
906 \ifcsundef{glossarymark}%
907 {%
908   \newcommand{\glsglossarymark}[1]{\glossarymark{#1}}
909 }%
910 {%
911   \@ifclassloaded{memoir}
912   {%
913     \newcommand{\glsglossarymark}[1]{%
914       \ifglsucmark
915         \markboth{\memUHead{#1}}{\memUHead{#1}}%
916       \else
917         \markboth{#1}{#1}%
918       \fi
919     }
920   }%
921   {%
922     \newcommand{\glsglossarymark}[1]{%
923       \ifglsucmark
924         \@mkboth{\mfirstucMakeUppercase{#1}}{\mfirstucMakeUppercase{#1}}%
925       \else
926         \@mkboth{#1}{#1}%
927       \fi
928     }
929   }
930 }

\glossarymark Provided for backward compatibility:

```

```

931 \providecommand{\glossarymark}[1]{%
932   \ifglsucmark
933     \@mkboth{\mfirstucMakeUppercase{#1}}{\mfirstucMakeUppercase{#1}}%
934   \else
935     \@mkboth{#1}{#1}%
936   \fi
937 }

```

The required sectional unit is given by `\@glossarysec` which was defined by the section package option. The starred form of the command is chosen. If you don't want any sectional command, you will need to redefine `\glossarysection`. The sectional unit can be changed, if different sectional units are required.

`\setglossarysection`

```

938 \newcommand*\setglossarysection[1]{%
939 \setkeys{glossaries.sty}{section=#1}}

```

The command `\@glossarysection` indicates how to start the glossary section if `\phantomsection` is not defined.

`\@glossarysection`

```

940 \newcommand*\@glossarysection[2]{%
941   \ifdefempty\@glossarysecstar
942   {%
943     \csname\@glossarysec\endcsname[#1]{#2}%
944   }%
945   {%
946     \csname\@glossarysec\endcsname*{#2}%
947     \@gls@toc{#1}{\@glossarysec}%
948   }%

```

Do automatic labelling if required

```

949   \@glossaryseclabel
950 }

```

As `\@glossarysection`, but put in `\phantomsection`, and swap where `\@gls@toc` goes. If using chapters do a `\clearpage`. This ensures that the hyper link from the table of contents leads to the line above the heading, rather than the line below it.

`\@pglossarysection`

```

951 \newcommand*\@pglossarysection[2]{%
952   \glsclearpage
953   \phantomsection
954   \ifdefempty\@glossarysecstar
955   {%
956     \csname\@glossarysec\endcsname{#2}%
957   }%
958   {%

```

```

959 \gls@toc{#1}{\glossarysec}%
960 \csname\glossarysec\endcsname*{#2}%
961 }%
    Do automatic labelling if required
962 \glossaryseclabel
963 }

```

`\gls@doclearpage` The `\gls@doclearpage` command is used to issue a `\clearpage` (or `\cleardoublepage`) depending on whether the glossary sectional unit is a chapter. If the sectional unit is something else, do nothing.

```

964 \newcommand*\gls@doclearpage{%
965 \ifthenelse{\equal{\glossarysec}{chapter}}{
966 {
967 \ifcsundef{cleardoublepage}{
968 {
969 \clearpage
970 }
971 {
972 \ifcsdef{if@openright}{
973 {
974 \if@openright
975 \cleardoublepage
976 \else
977 \clearpage
978 \fi
979 }
980 {
981 \cleardoublepage
982 }
983 }
984 }
985 }%
986 }

```

`\glsclearpage` This just calls `\gls@doclearpage`, but it makes it easier to have a user command so that the user can override it.

```

987 \newcommand*\glsclearpage{\gls@doclearpage}

```

The glossary is added to the table of contents if `glstoc` flag set. If it is set, `\gls@toc` will add a line to the `.toc` file, otherwise it will do nothing. (The first argument to `\gls@toc` is the title for the table of contents, the second argument is the sectioning type.)

`\gls@toc`

```

988 \newcommand*\gls@toc[2]{
989 \ifglstoc
990 \ifglslnumberline
991 \addcontentsline{toc}{#2}{\numberline{#1}}

```

```

992     \else
993         \addcontentsline{toc}{#2}{#1}%
994     \fi
995 \fi
996 }

```

1.4 Xindy

This section defines commands that only have an effect if xindy is used to sort the glossaries.

`\glsnoxindywarning` Issues a warning if xindy hasn't been specified. These warnings can be suppressed by redefining `\glsnoxindywarning` to ignore its argument

```

997 \newcommand*{\glsnoxindywarning}[1]{%
998   \GlossariesWarning{Not in xindy mode --- ignoring \string#1}%
999 }

```

`\@xdyattributes` Define list of attributes (`\string` is used in case the double quote character has been made active)

```

1000 \ifglsxindy
1001   \edef\@xdyattributes{\string"default\string"}%
1002 \fi

```

`\@xdyattributelist` Comma-separated list of attributes.

```

1003 \ifglsxindy
1004   \edef\@xdyattributelist{}%
1005 \fi

```

`\@xdylocref` Define list of markup location references.

```

1006 \ifglsxindy
1007   \def\@xdylocref{}
1008 \fi

```

`\@gls@ifinlist`

```

1009 \newcommand*{\@gls@ifinlist}[4]{%
1010   \def\@do@ifinlist##1,#1,##2\end@do@ifinlist{%
1011     \def\@gls@listsuffix{##2}%
1012     \ifx\@gls@listsuffix\@empty
1013       #4%
1014     \else
1015       #3%
1016     \fi
1017   }%
1018   \@do@ifinlist,#2,#1,\end@do@ifinlist
1019 }

```

`\GlsAddXdyCounters` Need to know all the counters that will be used in location numbers for Xindy. Argument may be a single counter name or a comma-separated list of counter names.

```

1020 \ifglxsindy
1021   \newcommand*{\@xdycounters}{\glscounter}
1022   \newcommand*\GlsAddXdyCounters[1]{%
1023     \@for\@gls@ctr:=#1\do{%
      Check if already in list before adding.
1024       \edef\@do@addcounter{%
1025         \noexpand\@gls@ifinlist{\@gls@ctr}{\@xdycounters}{}%
1026         {%
1027           \noexpand\edef\noexpand\@xdycounters{\@xdycounters,%
1028             \noexpand\@gls@ctr}%
1029         }%
1030       }%
1031       \@do@addcounter
1032     }
1033   }

Only has an effect before \writeist:
1034   \@onlypremakeg\GlsAddXdyCounters
1035 \else
1036   \newcommand*\GlsAddXdyCounters[1]{%
1037     \glsnoxindywarning\GlsAddXdyAttribute
1038   }
1039 \fi

```

`d@glssaddxdycounters` Counters must all be identified before adding attributes.

```

1040 \newcommand*\@disabled@glssaddxdycounters{%
1041   \PackageError{glossaries}{\string\GlsAddXdyCounters\space
1042     can't be used after \string\GlsAddXdyAttribute}{Move all
1043     occurrences of \string\GlsAddXdyCounters\space before the first
1044     instance of \string\GlsAddXdyAttribute}%
1045 }

```

`\GlsAddXdyAttribute` Adds an attribute.

```

1046 \ifglxsindy

First define internal command that adds an attribute for a given counter (2nd
argument is the counter):
1047   \newcommand*\@glssaddxdyattribute[2]{%

Add to xindy attribute list
1048     \edef\@xdyattributes{\@xdyattributes ^^J \string"#1\string" ^^J
1049       \string"#2#1\string"}%

Add to xindy markup location.
1050     \expandafter\toks@\expandafter{\@xdylocref}%
1051     \edef\@xdylocref{\the\toks@ ^^J%
1052       (markup-locref
1053       :open \string"\string~n%
1054       \expandafter\string\csname glsX#2X#1\endcsname
1055       \string" ^^J

```



```

1056      :close \string"\string" ^^J
1057      :attr \string"#2#1\string"))}%

Define associated attribute command \glsX<counter>X<attribute>{\Hprefix}{\<n>}
1058      \expandafter\gdef\csname glsX#2X#1\endcsname##1##2{%
1059          \setentrycounter[##1]{#2}\csname #1\endcsname{##2}%
1060      }%
1061  }

High-level command:
1062  \newcommand*\GlsAddXdyAttribute[1]{%

Add to comma-separated attribute list
1063      \ifx\@xdyattributelist\@empty
1064          \edef\@xdyattributelist{#1}%
1065      \else
1066          \edef\@xdyattributelist{\@xdyattributelist,#1}%
1067      \fi

Iterate through all specified counters and add counter-dependent attributes:
1068      \@for\@this@counter:=\@xdycounters\do{%
1069          \protected@edef\gls@do@addxdyattribute{%
1070              \noexpand\@glsaddxdyattribute{#1}{\@this@counter}%
1071          }
1072          \gls@do@addxdyattribute
1073      }%

All occurrences of \GlsAddXdyCounters must be used before this command
1074      \let\GlsAddXdyCounters\@disabled@glsaddxdycounters
1075  }

Only has an effect before \writeist:
1076  \@onlypremakeg\GlsAddXdyAttribute
1077 \else
1078  \newcommand*\GlsAddXdyAttribute[1]{%
1079      \glsnoxindywarning\GlsAddXdyAttribute}
1080 \fi

```

redefinedattributes Add known attributes for all defined counters

```

1081 \ifglxsindy
1082 \newcommand*\@gls@addpredefinedattributes{%
1083     \GlsAddXdyAttribute{glsnumberformat}
1084     \GlsAddXdyAttribute{textrm}
1085     \GlsAddXdyAttribute{textsf}
1086     \GlsAddXdyAttribute{texttt}
1087     \GlsAddXdyAttribute{textbf}
1088     \GlsAddXdyAttribute{textmd}
1089     \GlsAddXdyAttribute{textit}
1090     \GlsAddXdyAttribute{textup}
1091     \GlsAddXdyAttribute{textsl}
1092     \GlsAddXdyAttribute{textsc}
1093     \GlsAddXdyAttribute{emph}

```

```

1094 \GlsAddXdyAttribute{glshypernumber}
1095 \GlsAddXdyAttribute{hyperrrm}
1096 \GlsAddXdyAttribute{hypersf}
1097 \GlsAddXdyAttribute{hypertt}
1098 \GlsAddXdyAttribute{hyperbf}
1099 \GlsAddXdyAttribute{hypermd}
1100 \GlsAddXdyAttribute{hyperit}
1101 \GlsAddXdyAttribute{hyperup}
1102 \GlsAddXdyAttribute{hypersl}
1103 \GlsAddXdyAttribute{hypersc}
1104 \GlsAddXdyAttribute{hyperemph}
1105 }
1106 \else
1107 \let\@gls@addpredefinedattributes\relax
1108 \fi

```

`\@xdyuseralphabets` List of additional alphabets

```

1109 \def\@xdyuseralphabets{}

```

`\GlsAddXdyAlphabet` `\GlsAddXdyAlphabet{<name>}{<definition>}` adds a new alphabet called `<name>`. The definition must use xindy syntax.

```

1110 \ifglsxindy
1111 \newcommand*{\GlsAddXdyAlphabet}[2]{%
1112 \edef\@xdyuseralphabets{%
1113 \@xdyuseralphabets ^^J
1114 (define-alphabet "#1" (#2))}}
1115 \else
1116 \newcommand*{\GlsAddXdyAlphabet}[2]{%
1117 \glsnoxindywarning\GlsAddXdyAlphabet}
1118 \fi

```

This code is only required for xindy:

```

1119 \ifglsxindy

```

`\@xdy@locationlist` List of predefined location names.

```

1120 \newcommand*{\@gls@xdy@locationlist}{%
1121 roman-page-numbers,%
1122 Roman-page-numbers,%
1123 arabic-page-numbers,%
1124 alpha-page-numbers,%
1125 Alpha-page-numbers,%
1126 Appendix-page-numbers,%
1127 arabic-section-numbers%
1128 }

```

Each location class `<name>` has the format stored in `\@gls@xdy@Lclass@<name>`. Set up predefined formats.

@roman-page-numbers Lower case Roman numerals (i, ii, ...). In the event that \roman has been redefined to produce a fancy form of roman numerals, attempt to work out how it will be written to the output file.

```

1129 \protected@edef\@gls@roman{\@roman{0}\string"
1130 \string"roman-numbers-lowercase\string" :sep \string"}}%
1131 \@onelevel@sanitize\@gls@roman
1132 \edef\@tmp{\string" \string"roman-numbers-lowercase\string"
1133 :sep \string"}}%
1134 \@onelevel@sanitize\@tmp
1135 \ifx\@tmp\@gls@roman
1136 \expandafter
1137 \edef\csname @gls@xdy@Lclass@roman-page-numbers\endcsname{%
1138 \string"roman-numbers-lowercase\string"%
1139 }%
1140 \else
1141 \expandafter
1142 \edef\csname @gls@xdy@Lclass@roman-page-numbers\endcsname{
1143 :sep \string"\@gls@roman\string"%
1144 }%
1145 \fi

```

@Roman-page-numbers Upper case Roman numerals (I, II, ...).

```

1146 \expandafter\def\csname @gls@xdy@Lclass@Roman-page-numbers\endcsname{%
1147 \string"roman-numbers-uppercase\string"%
1148 }%

```

arabic-page-numbers Arabic numbers (1, 2, ...).

```

1149 \expandafter\def\csname @gls@xdy@Lclass@arabic-page-numbers\endcsname{%
1150 \string"arabic-numbers\string"%
1151 }%

```

@alpha-page-numbers Lower case alphabetical (a, b, ...).

```

1152 \expandafter\def\csname @gls@xdy@Lclass@alpha-page-numbers\endcsname{%
1153 \string"alpha\string"%
1154 }%

```

@Alpha-page-numbers Upper case alphabetical (A, B, ...).

```

1155 \expandafter\def\csname @gls@xdy@Lclass@Alpha-page-numbers\endcsname{%
1156 \string"ALPHA\string"%
1157 }%

```

pendix-page-numbers Appendix style locations (e.g. A-1, A-2, ..., B-1, B-2, ...). The separator is given by \@glsAlphacompositor.

```

1158 \expandafter\def\csname @gls@xdy@Lclass@Appendix-page-numbers\endcsname{%
1159 \string"ALPHA\string"
1160 :sep \string"\@glsAlphacompositor\string"
1161 \string"arabic-numbers\string"%
1162 }

```

arabic-section-numbers Section number style locations (e.g. 1.1, 1.2, ...). The compositor is given by
`\glscompositor.`

```

1163 \expandafter\def\csname @gls@xdy@Lclass@arabic-section-numbers\endcsname{%
1164   \string"arabic-numbers\string"
1165   :sep \string"\glscompositor\string"
1166   \string"arabic-numbers\string"%
1167 }%
```

xdyuserlocationdefs List of additional location definitions (separated by ^^J)

```

1168 \def\@xdyuserlocationdefs{}
```

xdyuserlocationnames List of additional user location names

```

1169 \def\@xdyuserlocationnames{}
```

End of xindy-only block:

```

1170 \fi
```

\GlsAddXdyLocation `\GlsAddXdyLocation[<prefix-loc>]{<name>}{<definition>}` Define a new location called *<name>*. The definition must use xindy syntax. (Note that this doesn't check to see if the location is already defined. That is left to xindy to complain about.)

```

1171 \ifglsxindy
1172   \newcommand*\GlsAddXdyLocation[3][{}]{%
1173     \def\@gls@tmp{#1}%
1174     \ifx\@gls@tmp\@empty
1175       \edef\@xdyuserlocationdefs{%
1176         \@xdyuserlocationdefs ^^J%
1177         (define-location-class \string"#2\string"^^J\space\space
1178         \space(:sep \string"{}\glsopenbrace\string" #3
1179         :sep \string"\glsclosebrace\string"))
1180       }%
1181     \else
1182       \edef\@xdyuserlocationdefs{%
1183         \@xdyuserlocationdefs ^^J%
1184         (define-location-class \string"#2\string"^^J\space\space
1185         \space(:sep "\glsopenbrace"
1186         #1
1187         :sep "\glsclosebrace\glsopenbrace" #3
1188         :sep "\glsclosebrace"))
1189       }%
1190     \fi
1191     \edef\@xdyuserlocationnames{%
1192       \@xdyuserlocationnames^^J\space\space\space
1193       \string"#1\string"}%
1194   }
```

Only has an effect before `\writeist`:

```

1195 \@onlypremakeg\GlsAddXdyLocation
```

```

1196 \else
1197   \newcommand*\GlsAddXdyLocation}[2]{%
1198     \glsnoxywarning\GlsAddXdyLocation}
1199 \fi

```

ylocationclassorder Define location class order

```

1200 \ifglxindy
1201   \edef\@xdylocationclassorder{^^J\space\space\space
1202     \string"roman-page-numbers\string"^^J\space\space\space
1203     \string"arabic-page-numbers\string"^^J\space\space\space
1204     \string"arabic-section-numbers\string"^^J\space\space\space
1205     \string"alpha-page-numbers\string"^^J\space\space\space
1206     \string"Roman-page-numbers\string"^^J\space\space\space
1207     \string"Alpha-page-numbers\string"^^J\space\space\space
1208     \string"Appendix-page-numbers\string"
1209     \@xdyuserlocationnames^^J\space\space\space
1210     \string"see\string"
1211   }
1212 \fi

```

Change the location order.

yLocationClassOrder

```

1213 \ifglxindy
1214   \newcommand*\GlsSetXdyLocationClassOrder[#1]{%
1215     \def\@xdylocationclassorder{#1}}
1216 \else
1217   \newcommand*\GlsSetXdyLocationClassOrder[#1]{%
1218     \glsnoxywarning\GlsSetXdyLocationClassOrder}
1219 \fi

```

\@xdysortrules Define sort rules

```

1220 \ifglxindy
1221   \def\@xdysortrules{}
1222 \fi

```

\GlsAddSortRule Add a sort rule

```

1223 \ifglxindy
1224   \newcommand*\GlsAddSortRule[#2]{%
1225     \expandafter\toks@\expandafter{\@xdysortrules}%
1226     \protected@edef\@xdysortrules{\the\toks@ ^^J
1227       (sort-rule \string"#1\string" \string"#2\string")}%
1228   }
1229 \else
1230   \newcommand*\GlsAddSortRule[#2]{%
1231     \glsnoxywarning\GlsAddSortRule}
1232 \fi

```

`\@xdyrequiredstyles` Define list of required styles (this should be a comma-separated list of xindy styles)

```

1233 \ifglxindy
1234   \def\@xdyrequiredstyles{tex}
1235 \fi

```

`\GlsAddXdyStyle` Add a xindy style to the list of required styles

```

1236 \ifglxindy
1237   \newcommand*\GlsAddXdyStyle[1]{%
1238     \edef\@xdyrequiredstyles{\@xdyrequiredstyles,#1}}%
1239 \else
1240   \newcommand*\GlsAddXdyStyle[1]{%
1241     \glsnnoxindywarning\GlsAddXdyStyle}
1242 \fi

```

`\GlsSetXdyStyles` Reset the list of required styles

```

1243 \ifglxindy
1244   \newcommand*\GlsSetXdyStyles[1]{%
1245     \edef\@xdyrequiredstyles{#1}}
1246 \else
1247   \newcommand*\GlsSetXdyStyles[1]{%
1248     \glsnnoxindywarning\GlsSetXdyStyles}
1249 \fi

```

`\findrootlanguage` This used to determine the root language, using a bit of trickery since babel doesn't supply the information, but now that babel is once again actively maintained, we can't do this any more, so `\findrootlanguage` is no longer available. Now provide a command that does nothing (in case it's been patched), but this may be removed completely in the future.

```

1250 \newcommand*\findrootlanguage{}

```

`\@xdylanguage` The xindy language setting is required by `makeglossaries`, so provide a command for `makeglossaries` to pick up the information from the auxiliary file. This command is not needed by the `glossaries` package, so define it to ignore its arguments.

```

1251 \def\@xdylanguage#1#2{}

```

`\GlsSetXdyLanguage` Define a command that allows the user to set the language for a given glossary type. The first argument indicates the glossary type. If omitted the main glossary is assumed.

```

1252 \ifglxindy
1253   \newcommand*\GlsSetXdyLanguage[2][\glsdefaulttype]{%
1254     \ifglossaryexists{#1}{%
1255       \expandafter\def\csname @xdy@#1@language\endcsname{#2}%
1256     }{%
1257       \PackageError{glossaries}{Can't set language type for
1258         glossary type '#1' --- no such glossary}{%

```

```

1259     You have specified a glossary type that doesn't exist}}
1260 \else
1261   \newcommand*\GlsSetXdyLanguage[2][]{%
1262     \glsnoxywarning\GlsSetXdyLanguage}
1263 \fi

```

`\@gls@codepage` The xindy codepage setting is required by makeglossaries, so provide a command for makeglossaries to pick up the information from the auxiliary file. This command is not needed by the glossaries package, so define it to ignore its arguments.

```

1264 \def\@gls@codepage#1#2{}

```

`\GlsSetXdyCodePage` Define command to set the code page.

```

1265 \ifglxindy
1266   \newcommand*\GlsSetXdyCodePage[1]{%
1267     \renewcommand*\@gls@codepage{#1}%
1268   }

```

Suggested by egreg:

```

1269   \AtBeginDocument{%
1270     \ifx\@gls@codepage\@empty
1271       \@ifpackageloaded{fontspec}{\def\@gls@codepage{utf8}}{}%
1272     \fi
1273   }
1274 \else
1275   \newcommand*\GlsSetXdyCodePage[1]{%
1276     \glsnoxywarning\GlsSetXdyCodePage}
1277 \fi

```

`\@xdylettergroups` Store letter group definitions.

```

1278 \ifglxindy
1279   \ifglxindy@glxnumbers
1280     \def\@xdylettergroups{(define-letter-group
1281       \string\glxnumbers\string^^J\space\space\space
1282       :prefixes (\string"0\string" \string"1\string"
1283       \string"2\string" \string"3\string" \string"4\string"
1284       \string"5\string" \string"6\string" \string"7\string"
1285       \string"8\string" \string"9\string")^^J\space\space\space
1286       :before \string"\@glxfirstletter\string")}
1287   \else
1288     \def\@xdylettergroups{}
1289   \fi
1290 \fi

```

`\GlsAddLetterGroup` Add a new letter group. The first argument is the name of the letter group. The second argument is the xindy code specifying prefixes and ordering.

```

1291   \newcommand*\GlsAddLetterGroup[2]{%
1292     \expandafter\toks@\expandafter{\@xdylettergroups}%
1293     \protected@edef\@xdylettergroups{\the\toks@^^J%

```

```

1294     (define-letter-group \string"#1\string"^^J\space\space\space#2))}%
1295 }%

```

1.5 Loops and conditionals

`\forall glossaries` To iterate through all glossaries (or comma-separated list of glossary names given in optional argument) use:

```
\forall glossaries[<glossary list>]{<cmd>}{<code>}
```

where *<cmd>* is a control sequence which will be set to the name of the glossary in the current iteration.

```

1296 \newcommand*\forallglossaries[3][\@glo@types]{%
1297   \@for#2:=#1\do{\ifx#2\@empty\else#3\fi}%
1298 }

```

`\forall acronyms`

```

1299 \newcommand*\forallacronyms[2]{%
1300   \@for#1:=\@glsacronymlists\do{\ifx#1\@empty\else#2\fi}%
1301 }

```

`\forall sentries` To iterate through all entries in a given glossary use:

```
\forall sentries[<type>]{<cmd>}{<code>}
```

where *<type>* is the glossary label and *<cmd>* is a control sequence which will be set to the entry label in the current iteration.

```

1302 \newcommand*\forallsentries[3][\glsdefaulttype]{%
1303   \edef\@glo@list{\csname glolist@#1\endcsname}%
1304   \@for#2:=\@glo@list\do
1305   {%
1306     \ifdefempty{#2}{-}{#3}%
1307   }%
1308 }

```

`\forall glsentries` To iterate through all glossary entries over all glossaries listed in the optional argument (the default is all glossaries) use:

```
\forall glsentries[<glossary list>]{<cmd>}{<code>}
```

Within `\forall glsentries`, the current glossary type is given by `\@thisglo@`.

```

1309 \newcommand*\forallglsentries[3][\@glo@types]{%
1310   \expandafter\forallglossaries\expandafter[#1]{\@thisglo@}%
1311   {%
1312     \forallsentries[\@thisglo@]{#2}{#3}%
1313   }%
1314 }

```


`\ifglossaryexists` To check to see if a glossary exists use:

```
\ifglossaryexists{<type>}{<true-text>}{<false-text>}
```

where `<type>` is the glossary's label.

```
1315 \newcommand{\ifglossaryexists}[3]{%
1316   \ifcsundef{@glo@#1@out}{#3}{#2}%
1317 }
```

Since the label is used to form the name of control sequences, by default UTF8 etc characters can't be used in the label. A possible workaround is to use `\scantokens`, but commands such as `\glentrytext` will no longer be usable in sectioning, caption etc commands. If the user really wants to be able to construct a label with UTF8 characters, allow them the means to do so (but on their own head be it, if they then use entries in `\section` etc). This can be done via:

```
\renewcommand*{\glsetoklabel}[1]{\scantokens{#1\noexpand}}
```

(Note, don't use `\detokenize` or it will cause commands like `\gladdall` to fail.) Since redefining `\glsetoklabel` can cause things to go badly wrong, I'm not going to mention it in the main user guide. Only advanced users who know what they're doing ought to attempt it.

`\glsetoklabel`

```
1318 \newcommand*{\glsetoklabel}[1]{#1}
```

`\ifglentryexists` To check to see if a glossary entry has been defined use:

```
\ifglentryexists{<label>}{<true text>}{<false text>}
```

where `<label>` is the entry's label.

```
1319 \newcommand{\ifglentryexists}[3]{%
1320   \ifcsundef{glo@\glsetoklabel{#1}@name}{#3}{#2}%
1321 }
```

`\ifglused` To determine if given glossary entry has been used in the document text yet use:

```
\ifglused{<label>}{<true text>}{<false text>}
```

where `<label>` is the entry's label. If true it will do `<true text>` otherwise it will do `<false text>`.

```
1322 \newcommand*{\ifglused}[3]{%
1323   \ifbool{glo@\glsetoklabel{#1}@flag}{#2}{#3}%
1324 }
```

The following two commands will cause an error if the given condition fails:

`\glsdoifexists` `\glsdoifexists{<label>}{<code>}`

Generate an error if entry specified by *<label>* doesn't exists, otherwise do *<code>*.

```
1325 \newcommand{\glsdoifexists}[2]{%
1326   \ifglentryexists{#1}{#2}{%
1327     \PackageError{glossaries}{Glossary entry ‘\glsdetoklabel{#1}’
1328       has not been defined}{You need to define a glossary entry before you
1329       can use it.}}%
1330 }
```

`\glsdoifnoexists` `\glsdoifnoexists{<label>}{<code>}`

The opposite: only do second argument if the entry doesn't exists. Generate an error message if it exists.

```
1331 \newcommand{\glsdoifnoexists}[2]{%
1332   \ifglentryexists{#1}{%
1333     \PackageError{glossaries}{Glossary entry ‘\glsdetoklabel{#1}’ has already
1334     been defined}{}}{#2}%
1335 }
```

`\glsdoifexistsorwarn` `\glsdoifexistsorwarn{<label>}{<code>}`

Generate a warning if entry specified by *<label>* doesn't exists, otherwise do *<code>*.

```
1336 \newcommand{\glsdoifexistsorwarn}[2]{%
1337   \ifglentryexists{#1}{#2}{%
1338     \GlossariesWarning{Glossary entry ‘\glsdetoklabel{#1}’
1339     has not been defined}%
1340   }%
1341 }
```

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

```
1342 \newcommand{\ifglshaschildren}[3]{%
1343   \glsdoifexists{#1}%
1344   {%
1345     \def\do@glshaschildren{#3}%
1346     \edef\@gls@thislabel{\glsdetoklabel{#1}}%
1347     \expandafter\forglentries\expandafter
1348     [\csname glo@\@gls@thislabel @type\endcsname]
1349     {\glo@label}%
1350     {%
1351       \letcs\glo@parent{glo@\glo@label @parent}%
1352       \ifdefequal\@gls@thislabel\glo@parent
1353       {%
1354         \def\do@glshaschildren{#2}%
1355         \@endfortrue
```

```

1356     }%
1357     {}%
1358     }%
1359     \do@glshaschildren
1360 }%
1361 }

```

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

```

1362 \newcommand{\ifglshasparent}[3]{%
1363   \glsdoifexists{#1}%
1364   {%
1365     \ifcsempy{glo@\glsdetoklabel{#1}@parent}{#3}{#2}%
1366   }%
1367 }

```

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

```

1368 \newcommand*{\ifglshasdesc}[3]{%
1369   \ifcsempy{glo@\glsdetoklabel{#1}@desc}%
1370   {#3}%
1371   {#2}%
1372 }

```

`\ifglsdessuppressed` `\ifglsdessuppressed{<label>}{<true part>}{<false part>}` Does *<true part>* if the description is just `\nopostdesc` otherwise does *<false part>*.

```

1373 \newcommand*{\ifglsdessuppressed}[3]{%
1374   \ifcsequal{glo@\glsdetoklabel{#1}@desc}{@no@post@desc}%
1375   {#2}%
1376   {#3}%
1377 }

```

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

```

1378 \newcommand*{\ifglshassymbol}[3]{%
1379   \letcs{\@glo@symbol}{glo@\glsdetoklabel{#1}@symbol}%
1380   \ifdefempty\@glo@symbol
1381   {#3}%
1382   {%
1383     \ifdefequal\@glo@symbol\@gls@default@value
1384     {#3}%
1385     {#2}%
1386   }%
1387 }

```

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

```

1388 \newcommand*{\ifglshaslong}[3]{%
1389   \letcs{\@glo@long}{glo@\glsdetoklabel{#1}@long}%

```

```

1390 \ifdefempty\@glo@long
1391   {#3}%
1392   {%
1393     \ifdefequal\@glo@long\@gls@default@value
1394     {#3}%
1395     {#2}%
1396   }%
1397 }

```

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

```

1398 \newcommand*{\ifglshasshort}[3]{%
1399   \letcs{\@glo@short}{glo\glsdetoklabel{#1}@short}%
1400   \ifdefempty\@glo@short
1401   {#3}%
1402   {%
1403     \ifdefequal\@glo@short\@gls@default@value
1404     {#3}%
1405     {#2}%
1406   }%
1407 }

```

`\ifglshasfield` `\ifglshasfield{<field>}{<label>}{<true part>}{<false part>}`

```

1408 \newcommand*{\ifglshasfield}[4]{%
1409   \glsdoifexists{#2}%
1410   {%
1411     \letcs{\@glo@thisvalue}{glo\glsdetoklabel{#2}@#1}%

```

First check supplied field label is defined.

```

1412   \ifdef\@glo@thisvalue
1413   {%

```

Is defined, so now check if empty.

```

1414     \ifdefempty\@glo@thisvalue
1415     {%

```

Is empty, so doesn't have field set.

```

1416       #4%
1417     }%
1418   {%

```

Not empty, so check if set to `\@gls@default@value`

```

1419     \ifdefequal\@glo@thisvalue\@gls@default@value{#4}{#3}%
1420     }%
1421   }%
1422 }%

```

Field given isn't defined, so check if mapping exists.

```

1423   \@gls@fetchfield{\@gls@thisfield}{#1}%

```

If `\@gls@thisfield` is defined, we've found a map. If not, the field supplied doesn't exist.

```
1424 \ifdef\@gls@thisfield
1425 {%
```

Is defined, so now check if empty.

```
1426 \letcs{\@glo@thisvalue}{glo\glsdetoklabel{#2}\@gls@thisfield}%
1427 \ifdefempty\@glo@thisvalue
1428 {%
```

Is empty so field hasn't been set.

```
1429 #4%
1430 }%
1431 {%
```

Isn't empty so check if it's been set to `\@gls@default@value`.

```
1432 \ifdequal\@glo@thisvalue\@gls@default@value{#4}{#3}%
1433 }%
1434 }%
1435 {%
```

Not defined.

```
1436 \GlossariesWarning{Unknown entry field '#1'}%
1437 #4%
1438 }%
1439 }%
1440 }%
1441 }
```

1.6 Defining new glossaries

A comma-separated list of glossary names is stored in `\@glo@types`. When a new glossary type is created, its identifying name is added to this list. This is used by commands that iterate through all glossaries (such as `\makeglossaries` and `\printglossaries`).

`\@glo@types`

```
1442 \newcommand*{\@glo@types}{,}
```

`\provide@newglossary` If the user removes the glossary package from their document, ensure the next run doesn't throw a load of undefined control sequence errors when the aux file is parsed.

```
1443 \newcommand*\@gls@provide@newglossary{%
1444 \protected@write\@auxout{}\string\providecommand\string\@newglossary[4]{}%

```

Only need to do this once.

```
1445 \let\@gls@provide@newglossary\relax
1446 }
```

`\defglentryfmt` Allow different glossaries to have different display styles.

```
1447 \newcommand*\defglentryfmt}[2][\glsdefaulttype]{%
1448   \csgdef{gls@#1@entryfmt}{#2}%
1449 }
```

`\gls@doentryfmt`

```
1450 \newcommand*\gls@doentryfmt}[1]{\csuse{gls@#1@entryfmt}}
```

`\@gls@forbidtext` As a security precaution, don't allow the user to specify a 'tex' extension for any of the glossary files. (Just in case a seriously confused novice user doesn't know what they're doing.) The argument must be a control sequence whose replacement text is the requested extension.

```
1451 \newcommand*\@gls@forbidtext}[1]{%
1452   \ifboolexpr{test {\ifdefstring{#1}{tex}}
1453     or test {\ifdefstring{#1}{TEX}}}
1454   {%
1455     \def#1{nottex}%
1456     \PackageError{glossaries}%
1457       {Forbidden '.tex' extension replaced with '.nottex'}%
1458       {I'm sorry, I can't allow you to do something so reckless.\MessageBreak
1459       Don't use '.tex' as an extension for a temporary file.}%
1460   }%
1461   {%
1462   }%
1463 }
```

A new glossary type is defined using `\newglossary`. Syntax:

```
\newglossary[<log-ext>]{<name>}{<in-ext>}{<out-ext>}
{<title>}[<counter>]
```

where *<log-ext>* is the extension of the makeindex transcript file, *<in-ext>* is the extension of the glossary input file (read in by `\printglossary` and created by makeindex), *<out-ext>* is the extension of the glossary output file which is read in by makeindex (lines are written to this file by the `\glossary` command), *<title>* is the title of the glossary that is used in `\glossarysection` and *<counter>* is the default counter to be used by entries belonging to this glossary. The `makeglossaries` Perl script reads in the relevant extensions from the auxiliary file, and passes the appropriate file names and switches to makeindex.

`\newglossary`

```
1464 \newcommand*\newglossary{\@ifstar\s@newglossary\ns@newglossary}
```

`\s@newglossary` The starred version will construct the extension based on the label.

```
1465 \newcommand*\s@newglossary}[2]{%
1466   \ns@newglossary[#1-glg]{#1}{#1-gls}{#1-glo}{#2}%
1467 }
```

`\ns@newglossary` Define the unstarred version.

```
1468 \newcommand*{\ns@newglossary}[5][glg]{%
1469   \ifglossaryexists{#2}%
1470   {%
1471     \PackageError{glossaries}{Glossary type ‘#2’ already exists}{%
1472       You can’t define a new glossary called ‘#2’ because it already
1473       exists}%
1474   }%
1475   {%
```

Check if default has been set

```
1476   \ifundef\glsdefaulttype
1477   {%
1478     \gdef\glsdefaulttype{#2}%
1479   }{}%
```

Add this to the list of glossary types:

```
1480   \toks@{#2}\edef\@glo@types{\@glo@types\the\toks@,}%
```

Define a comma-separated list of labels for this glossary type, so that all the entries for this glossary can be reset with a single command. When a new entry is created, its label is added to this list.

```
1481   \expandafter\gdef\csname glolist@#2\endcsname{,}%
```

Store the file extensions:

```
1482   \expandafter\edef\csname @glotype@#2@log\endcsname{#1}%
1483   \expandafter\edef\csname @glotype@#2@in\endcsname{#3}%
1484   \expandafter\edef\csname @glotype@#2@out\endcsname{#4}%
1485   \expandafter\@gls@forbidtexext\csname @glotype@#2@log\endcsname
1486   \expandafter\@gls@forbidtexext\csname @glotype@#2@in\endcsname
1487   \expandafter\@gls@forbidtexext\csname @glotype@#2@out\endcsname
```

Store the title:

```
1488   \expandafter\def\csname @glotype@#2@title\endcsname{#5}%
```

```
1489   \@gls@provide@newglossary
```

```
1490   \protected@write\auxout{}\string\@newglossary{#2}{#1}{#3}{#4}}%
```

How to display this entry in the document text (uses `\glsentry` by default).

This can be redefined by the user later if required (see `\defglsentry`). This may already have been defined if this has been specified as a list of acronyms.

```
1491   \ifcsundef{gls@#2@entryfmt}%
1492   {%
1493     \defglsentryfmt[#2]{\glsentryfmt}%
1494   }%
1495   {}%
```

Define sort counter if required:

```
1496   \@gls@defsortcount{#2}%
```

Find out if the final optional argument has been specified, and use it to set the counter associated with this glossary. (Uses `\glscounter` if no optional argument is present.)

```
1497 \ifnextchar[{\@gls@setcounter{#2}}%
1498   {\@gls@setcounter{#2}[\glscounter]}}%
1499 }
```

`\altnewglossary`

```
1500 \newcommand*{\altnewglossary}[3]{%
1501   \newglossary[#2-glg]{#1}{#2-gls}{#2-glo}{#3}%
1502 }
```

Only define new glossaries in the preamble:

```
1503 \@onlypreamble{\newglossary}
```

Only define new glossaries before `\makeglossaries`

```
1504 \@onlypremakeg\newglossary
```

`\newglossary` is used to specify the file extensions for the `makeindex` input, output and transcript files. It is written to the auxiliary file by `\newglossary`. Since it is not used by \LaTeX , `\newglossary` simply ignores its arguments.

`\@newglossary`

```
1505 \newcommand*{\@newglossary}[4]{}
```

Store counter to be used for given glossary type (the first argument is the glossary label, the second argument is the name of the counter):

`\@gls@setcounter`

```
1506 \def\@gls@setcounter#1[#2]{%
1507   \expandafter\def\csname @glotype@#1@counter\endcsname{#2}%
```

Add counter to xindy list, if not already added:

```
1508   \ifglsxindy
1509     \GlsAddXdyCounters{#2}%
1510   \fi
1511 }
```

Get counter associated with given glossary (the argument is the glossary label):

`\@gls@getcounter`

```
1512 \newcommand*{\@gls@getcounter}[1]{%
1513   \csname @glotype@#1@counter\endcsname
1514 }
```

Define the main glossary. This will be the first glossary to be displayed when using `\printglossaries`.

```
1515 \glsdefmain
```

Define the “acronym” glossaries if required.

```
1516 \@gls@do@acronymsdef
```


Define the “symbols”, “numbers” and “index” glossaries if required.

```
1517 \@gls@do@symbolsdef
1518 \@gls@do@numbersdef
1519 \@gls@do@indexdef
```

`\newignoredglossary` Creates a new glossary that doesn't have associated files. This glossary is ignored by and commands that iterate over glossaries, such as `\printglossaries`, and won't work with commands like `\printglossary`. It's intended for entries that are so commonly-known they don't require a glossary.

```
1520 \newcommand*{\newignoredglossary}[1]{%
1521   \ifdefempty\@ignored@glossaries
1522   {%
1523     \edef\@ignored@glossaries{#1}%
1524   }%
1525   {%
1526     \eappto\@ignored@glossaries{,#1}%
1527   }%
1528   \csgdef{glolist@#1}{,}%
1529   \ifcsundef{gls@#1@entryfmt}%
1530   {%
1531     \defglentryfmt[#1]{\glentryfmt}%
1532   }%
1533   {%
1534     \ifdefempty\@gls@nohyperlist
1535     {%
1536       \renewcommand*{\@gls@nohyperlist}{#1}%
1537     }%
1538     {%
1539       \eappto\@gls@nohyperlist{,#1}%
1540     }%
1541 }
```

`@ignored@glossaries` List of ignored glossaries.

```
1542 \newcommand*{\@ignored@glossaries}{}
```

`\ifignoredglossary` Tests if the given glossary is an ignored glossary. Expansion is used in case the first argument is a control sequence.

```
1543 \newcommand*{\ifignoredglossary}[3]{%
1544   \edef\@gls@igtype{#1}%
1545   \expandafter\DTLifinlist\expandafter
1546   {\@gls@igtype}{\@ignored@glossaries}{#2}{#3}%
1547 }
```

1.7 Defining new entries

New glossary entries are defined using `\newglossaryentry`. This command requires a label and a key-value list that defines the relevant information for that entry. The definition for these keys follows. Note that the name, description

and symbol keys will be sanitized later, depending on the value of the package option `sanitize` (this means that if some of the keys haven't been defined, they can be constructed from the name and description key before they are sanitized).

name The name key indicates the name of the term being defined. This is how the term will appear in the glossary. The name key is required when defining a new glossary entry.

```
1548 \define@key{glossentry}{name}{%
1549 \def\@glo@name{#1}%
1550 }
```

description The description key is usually only used in the glossary, but can be made to appear in the text by redefining `\glsentryfmt` or using `\defglsentryfmt`. The description key is required when defining a new glossary entry. If a long description is required, use `\longnewglossaryentry` instead of `\newglossaryentry`.

```
1551 \define@key{glossentry}{description}{%
1552 \def\@glo@desc{#1}%
1553 }
```

descriptionplural

```
1554 \define@key{glossentry}{descriptionplural}{%
1555 \def\@glo@descplural{#1}%
1556 }
```

sort The sort key needs to be sanitized here (the sort key is provided for `makeindex`'s benefit, not for use in the document). The sort key is optional when defining a new glossary entry. If omitted, the value is given by *<name>* *<description>*.

```
1557 \define@key{glossentry}{sort}{%
1558 \def\@glo@sort{#1}}
```

text The text key determines how the term should appear when used in the document (i.e. outside of the glossary). If omitted, the value of the name key is used instead.

```
1559 \define@key{glossentry}{text}{%
1560 \def\@glo@text{#1}%
1561 }
```

plural The plural key determines how the plural form of the term should be displayed in the document. If omitted, the plural is constructed by appending `\glspluralsuffix` to the value of the text key.

```
1562 \define@key{glossentry}{plural}{%
1563 \def\@glo@plural{#1}%
1564 }
```

first The first key determines how the entry should be displayed in the document when it is first used. If omitted, it is taken to be the same as the value of the text key.

```
1565 \define@key{glossentry}{first}{%
1566 \def\@glo@first{#1}%
1567 }
```

firstplural The firstplural key is used to set the plural form for first use, in the event that the plural is required the first time the term is used. If omitted, it is constructed by appending `\glspluralsuffix` to the value of the first key.

```
1568 \define@key{glossentry}{firstplural}{%
1569 \def\@glo@firstplural{#1}%
1570 }
```

`\@gls@default@value`

```
1571 \newcommand*{\@gls@default@value}{\relax}
```

symbol The symbol key is ignored by most of the predefined glossary styles, and defaults to `\relax` if omitted. It is provided for glossary styles that require an associated symbol, as well as a name and description. To make this value appear in the glossary, you need to redefine `\glossentry`. If you want this value to appear in the text when the term is used by commands like `\gls`, you will need to change `\glsentryfmt` (or use for `\defglsentryfmt` individual glossaries).

```
1572 \define@key{glossentry}{symbol}{%
1573 \def\@glo@symbol{#1}%
1574 }
```

symbolplural

```
1575 \define@key{glossentry}{symbolplural}{%
1576 \def\@glo@symbolplural{#1}%
1577 }
```

type The type key specifies to which glossary this entry belongs. If omitted, the default glossary is used.

```
1578 \define@key{glossentry}{type}{%
1579 \def\@glo@type{#1}}
```

counter The counter key specifies the name of the counter associated with this glossary entry:

```
1580 \define@key{glossentry}{counter}{%
1581 \ifcsundef{c@#1}%
1582 {%
1583 \PackageError{glossaries}%
1584 {There is no counter called ‘#1’}%
1585 {%
1586 The counter key should have the name of a valid counter
1587 as its value%
```

```

1588 }%
1589 }%
1590 {%
1591   \def\@glo@counter{#1}%
1592 }%
1593 }

```

see The see key specifies a list of cross-references

```

1594 \define@key{glossentry}{see}{%
1595   \gls@checkseeallowed
1596   \def\@glo@see{#1}%
1597   \@glo@seeautonumberlist
1598 }

```

gls@checkseeallowed

```

1599 \newcommand*{\gls@checkseeallowed}{%
1600   \PackageError{glossaries}%
1601   {'see' key may only be used after \string\makeglossaries\space
1602   or \string\makenoidxglossaries}%
1603   {You must use \string\makeglossaries\space
1604   or \string\makenoidxglossaries\space before defining
1605   any entries that have a 'see' key}%
1606 }

```

parent The parent key specifies the parent entry, if required.

```

1607 \define@key{glossentry}{parent}{%
1608   \def\@glo@parent{#1}}

```

nonumberlist The nonumberlist key suppresses or activates the number list for the given entry.

```

1609 \define@choicekey{glossentry}{nonumberlist}[\val\nr]{true,false}[true]{%
1610   \ifcase\nr\relax
1611     \def\@glo@prefix{\glsnonextpages}%
1612   \else
1613     \def\@glo@prefix{\glsnextpages}%
1614   \fi
1615 }

```

Define some generic user keys. (Additional keys can be added by the user.)

user1

```

1616 \define@key{glossentry}{user1}{%
1617   \def\@glo@useri{#1}%
1618 }

```

user2

```

1619 \define@key{glossentry}{user2}{%
1620   \def\@glo@userii{#1}%
1621 }

```

user3

```
1622 \define@key{glossentry}{user3}{%  
1623   \def\@glo@useriii{#1}%  
1624 }
```

user4

```
1625 \define@key{glossentry}{user4}{%  
1626   \def\@glo@useriv{#1}%  
1627 }
```

user5

```
1628 \define@key{glossentry}{user5}{%  
1629   \def\@glo@userv{#1}%  
1630 }
```

user6

```
1631 \define@key{glossentry}{user6}{%  
1632   \def\@glo@uservi{#1}%  
1633 }
```

short This key is provided for use by \newacronym. It's not designed for general purpose use, so isn't described in the user manual.

```
1634 \define@key{glossentry}{short}{%  
1635   \def\@glo@short{#1}%  
1636 }
```

shortplural This key is provided for use by \newacronym.

```
1637 \define@key{glossentry}{shortplural}{%  
1638   \def\@glo@shortpl{#1}%  
1639 }
```

long This key is provided for use by \newacronym.

```
1640 \define@key{glossentry}{long}{%  
1641   \def\@glo@long{#1}%  
1642 }
```

longplural This key is provided for use by \newacronym.

```
1643 \define@key{glossentry}{longplural}{%  
1644   \def\@glo@longpl{#1}%  
1645 }
```

\@glsnname Define command to generate error if name key is missing.

```
1646 \newcommand*\@glsnname{%  
1647   \PackageError{glossaries}{name key required in  
1648   \string\newglossaryentry\space for entry '\@glo@label'}{You  
1649   haven't specified the entry name}}
```

`\@glsnodesc` Define command to generate error if description key is missing.

```
1650 \newcommand*\@glsnodesc{%
1651   \PackageError{glossaries}
1652   {%
1653     description key required in \string\newglossaryentry\space
1654     for entry '\@glo@label'%
1655   }%
1656   {%
1657     You haven't specified the entry description%
1658   }%
1659 }%
```

`\@glsdefaultplural` Now obsolete. Don't use.

```
1660 \newcommand*\@glsdefaultplural{}
```

`\@gls@missingnumberlist` Define a command to generate warning when numberlist not set.

```
1661 \newcommand*\@gls@missingnumberlist}[1]{%
1662   ??%
1663   \ifglssavenumberlist
1664     \GlossariesWarning{Missing number list for entry '#1'.
1665       Maybe makeglossaries + rerun required.}%
1666   \else
1667     \PackageError{glossaries}%
1668     {Package option 'savenumberlist=true' required.}%
1669     {%
1670       You must use the 'savenumberlist' package option
1671       to reference location lists.%
1672     }%
1673   \fi
1674 }
```

`\@glsdefaultsort` Define command to set default sort.

```
1675 \newcommand*\@glsdefaultsort{\@glo@name}
```

`\gls@level` Register to increment entry levels.

```
1676 \newcount\gls@level
```

`\@gls@noexpand@field`

```
1677 \newcommand*\@gls@noexpand@field}[3]{%
1678   \expandafter\global\expandafter
1679   \let\csname glo@#1@#2\endcsname#3%
1680 }
```

`\gls@noexpand@fields`

```
1681 \newcommand*\@gls@noexpand@fields}[4]{%
1682   \ifcsdef{gls@assign@#3@field}
1683   {%
1684     \ifdefequal{#4}{\@gls@default@value}%
```

```

1685     {%
1686         \edef\@gls@value{\expandonce{#1}}%
1687         \csuse{gls@assign@#3@field}{#2}{\@gls@value}%
1688     }%
1689     {%
1690         \csuse{gls@assign@#3@field}{#2}{#4}%
1691     }%
1692 }%
1693 {%
1694     \ifdefequal{#4}{\@gls@default@value}%
1695     {%
1696         \edef\@gls@value{\expandonce{#1}}%
1697         \@gls@noexpand@field{#2}{#3}{\@gls@value}%
1698     }%
1699     {%
1700         \@gls@noexpand@field{#2}{#3}{#4}%
1701     }%
1702 }%
1703 }

```

\@gls@expand@field

```

1704 \newcommand{\@gls@expand@field}[3]{%
1705   \expandafter
1706   \protected@xdef\csname glo@#1@#2\endcsname{#3}%
1707 }

```

@gls@expand@fields

```

1708 \newcommand{\@gls@expand@fields}[4]{%
1709   \ifcsdef{gls@assign@#3@field}
1710   {%
1711       \ifdefequal{#4}{\@gls@default@value}%
1712       {%
1713           \edef\@gls@value{\expandonce{#1}}%
1714           \csuse{gls@assign@#3@field}{#2}{\@gls@value}%
1715       }%
1716       {%
1717           \expandafter\@gls@startswitexpandonce#4\relax\relax\gls@endcheck
1718           {%
1719               \@gls@expand@field{#2}{#3}{#4}%
1720           }%
1721           {%
1722               \csuse{gls@assign@#3@field}{#2}{#4}%
1723           }%
1724       }%
1725   }%
1726   {%
1727       \ifdefequal{#4}{\@gls@default@value}%
1728       {%

```

```

1729      \@gls@expand@field{#2}{#3}{#1}%
1730    }%
1731    {%
1732      \@gls@expand@field{#2}{#3}{#4}%
1733    }%
1734  }%
1735 }

```

startswithexpandonce

```

1736 \def\@gls@expandonce{\expandonce}
1737 \def\@gls@startswithexpandonce#1#2\gls@endcheck#3#4{%
1738   \def\@gls@tmp{#1}%
1739   \ifdefequal{\@gls@expandonce}{\@gls@tmp}{#3}{#4}%
1740 }

```

`\gls@assign@field` `\gls@assign@field{<def value>}{<glossary type>}{<field>}{<tmp cs>}`

Assigns an entry field. Expansion performed by default (except for name, symbol and description where backward compatibility required). If `<tmp cs>` is `<@gls@default@value>`, `<def value>` is used instead.

```

1741 \let\gls@assign@field\@gls@expand@fields

```

`\glsexpandfields` Fully expand values when assigning fields (except for specific fields that are overridden by `\glssetnoexpandfield`).

```

1742 \newcommand*\glsexpandfields{%
1743   \let\gls@assign@field\@gls@expand@fields
1744 }

```

`\glsnoexpandfields` Don't expand values when assigning fields (except for specific fields that are overridden by `\glssetexpandfield`).

```

1745 \newcommand*\glsnoexpandfields{%
1746   \let\gls@assign@field\@gls@noexpand@fields
1747 }

```

`\newglossaryentry` Define `\newglossaryentry {<label>}{<key-val list>}`. There are two required fields in `<key-val list>`: name (or parent) and description. (See above.)

```

1748 \newrobustcmd{\newglossaryentry}[2]{%

```

Check to see if this glossary entry has already been defined:

```

1749   \glsdoifnoexists{#1}%
1750   {%
1751     \gls@defglossaryentry{#1}{#2}%
1752   }%
1753 }

```

`\provideglossaryentry` Like `\newglossaryentry` but does nothing if the entry has already been defined.


```

1754 \newrobustcmd{\provideglossaryentry}[2]{%
1755   \ifglentryexists{#1}%
1756   {}%
1757   {%
1758     \gls@defglossaryentry{#1}{#2}%
1759   }%
1760 }
1761 \@onlypreamble{\provideglossaryentry}

```

\newglossaryentry For use in document environment.

```

1762 \newrobustcmd{\newglossaryentry}[2]{%
1763   \ifundef\@gls@deffile
1764   {%
1765     \global\newwrite\@gls@deffile
1766     \immediate\openout\@gls@deffile=\jobname.glsdefs
1767   }%
1768   {}%
1769   \ifglentryexists{#1}{}%
1770   {%
1771     \gls@defglossaryentry{#1}{#2}%
1772   }%
1773   \@gls@writedef{#1}%
1774 }
1775 \AtBeginDocument
1776 {
1777   \makeatletter
1778   \InputIfFileExists{\jobname.glsdefs}{%}{%}
1779   \makeatother
1780   \let\newglossaryentry\newglossaryentry
1781 }
1782 \AtEndDocument{\ifdef\@gls@deffile{\closeout\@gls@deffile}{%}}

```

\@gls@writedef Writes glossary entry definition to \@gls@deffile.

```

1783 \newcommand*{\@gls@writedef}[1]{%
1784   \immediate\write\@gls@deffile
1785   {%
1786     \string\ifglentryexists{#1}{%\expandafter\@gobble\string\%^~J%
1787     \expandafter\@gobble\string\{\expandafter\@gobble\string\%^~J%
1788     \string\gls@defglossaryentry{\glsdetoklabel{#1}}\expandafter
1789     \@gobble\string\%^~J%
1790     \expandafter\@gobble\string\{\expandafter\@gobble\string\%%
1791   }%

```

Write key value information:

```

1792 \@for\@gls@map:=\@gls@keymap\do
1793 {%
1794   \edef\glo@value{\expandafter\expandonce
1795     \csname glo@\glsdetoklabel{#1}\@expandafter
1796     \@secondoftwo\@gls@map\endcsname}%
1797   \@onelevel@sanitize\glo@value

```

```

1798 \immediate\write\@gls@deffile
1799 {%
1800 \expandafter\@firstoftwo\@gls@map
1801 =\expandafter\@gobble\string\{\glo@value\expandafter\@gobble\string\},%
1802 \expandafter\@gobble\string\%%
1803 }%
1804 }%

Provide hook:

1805 \glswritedefhook
1806 \immediate\write\@gls@deffile
1807 {%
1808 \expandafter\@gobble\string\%^~J%
1809 \expandafter\@gobble\string\}\expandafter\@gobble\string\%^~J%
1810 \expandafter\@gobble\string\}\expandafter\@gobble\string\%%
1811 }%
1812 }

```

`\@gls@keymap` List of entry definition key names and corresponding tag in control sequence used to store the value.

```

1813 \newcommand*{\@gls@keymap}{%
1814 {name}{name},%
1815 {sort}{sortvalue},% unescaped sort value
1816 {type}{type},%
1817 {first}{first},%
1818 {firstplural}{firstpl},%
1819 {text}{text},%
1820 {plural}{plural},%
1821 {description}{desc},%
1822 {descriptionplural}{descplural},%
1823 {symbol}{symbol},%
1824 {symbolplural}{symbolplural},%
1825 {user1}{useri},%
1826 {user2}{userii},%
1827 {user3}{useriii},%
1828 {user4}{useriv},%
1829 {user5}{userv},%
1830 {user6}{uservi},%
1831 {long}{long},%
1832 {longplural}{longpl},%
1833 {short}{short},%
1834 {shortplural}{shortpl},%
1835 {counter}{counter},%
1836 {parent}{parent}%
1837 }

```

`\@gls@fetchfield` `\@gls@fetchfield{<cs>}{<field>}`

Fetches the internal field label from the given user *<field>* and stores in *<cs>*.

```
1838 \newcommand*\@gls@fetchfield}[2]{%
```

Ensure user field name is fully expanded

```
1839 \edef\@gls@thisval{#2}%
```

Iterate through known mappings until we find the one for this field.

```
1840 \for\@gls@map:=\@gls@keymap\do{%
```

```
1841 \edef\@this@key{\expandafter\@firstoftwo\@gls@map}%
```

```
1842 \ifdefequal{\@this@key}{\@gls@thisval}%
```

```
1843 {%
```

Found it.

```
1844 \edef#1{\expandafter\@secondoftwo\@gls@map}%
```

Break out of loop.

```
1845 \@endfortrue
```

```
1846 }%
```

```
1847 {}%
```

```
1848 }%
```

```
1849 }
```

```
\glsaddkey \glsaddkey{<key>}{<default value>}{<no link cs>}{<no link ucfirst cs>}{<link cs>}{<link ucfirst cs>}{<link allcaps cs>}
```

Allow user to add their own custom keys.

```
1850 \newcommand*\@glsaddkey{\@ifstar\@sglsaddkey\@glsaddkey}
```

Starred version switches on expansion for this key.

```
1851 \newcommand*\@sglsaddkey}[1]{%
```

```
1852 \key@ifundefined{glossentry}{#1}%
```

```
1853 {%
```

```
1854 \expandafter\newcommand\expandafter*\expandafter
```

```
1855 {\csname gls@assign@#1@field\endcsname}[2]{%
```

```
1856 \@gls@expand@field{##1}{#1}{##2}%
```

```
1857 }%
```

```
1858 }%
```

```
1859 {}%
```

```
1860 \@glsaddkey{#1}%
```

```
1861 }
```

Unstarred version doesn't override default expansion.

```
1862 \newcommand*\@glsaddkey}[7]{%
```

Check the specified key doesn't already exist.

```
1863 \key@ifundefined{glossentry}{#1}%
```

```
1864 {%
```

Set up the key.

```
1865 \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
```

```
1866 \appto\@gls@keymap{, #1}{#1}}%
```

Set the default value.

```
1867 \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
```

Assignment code.

```
1868 \appto\@newglossaryentryposthook{%
1869 \letcs{\@glo@tmp}{@glo@#1}%
1870 \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
1871 }%
```

Define the no-link commands.

```
1872 \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
1873 \newcommand*{#4}[1]{\@Gls@entry@field{##1}{#1}}%
```

Now for the commands with links. First the version with no case change:

```
1874 \ifcsdef{@gls@user@#1@}%
1875 {%
1876 \PackageError{glossaries}%
1877 {Can't define '\string#5' as helper command
1878 '\expandafter\string\csname @gls@user@#1@\endcsname' already exists}%
1879 }%
1880 }%
1881 {%
1882 \newrobustcmd*{#5}{\@gls@hyp@opt{\csuse{@gls@user@#1}}}%
1883 \expandafter\newcommand\expandafter*\expandafter
1884 {\csname @gls@user@#1\endcsname}[2][]{%
1885 \new@ifnextchar[%
1886 {\csuse{@gls@user@#1@}{##1}{##2}}%
1887 {\csuse{@gls@user@#1@}{##1}{##2}[]}}%
1888 \csdef{@gls@user@#1@}##1##2[##3]{%
1889 \@gls@field@link{##1}{##2}{#3{##2}##3}%
1890 }%
1891 }%
```

Next the version with the first letter converted to upper case:

```
1892 \ifcsdef{@Gls@user@#1@}%
1893 {%
1894 \PackageError{glossaries}%
1895 {Can't define '\string#6' as helper command
1896 '\expandafter\string\csname @Gls@user@#1@\endcsname' already exists}%
1897 }%
1898 }%
1899 {%
1900 \newrobustcmd*{#6}{\@gls@hyp@opt{\csuse{@Gls@user@#1}}}%
1901 \expandafter\newcommand\expandafter*\expandafter
1902 {\csname @Gls@user@#1\endcsname}[2][]{%
1903 \new@ifnextchar[%
1904 {\csuse{@Gls@user@#1@}{##1}{##2}}%
1905 {\csuse{@Gls@user@#1@}{##1}{##2}[]}}%
1906 \csdef{@Gls@user@#1@}##1##2[##3]{%
```

```

1907      \@gls@field@link{##1}{##2}{#4{##2}##3}%
1908    }%
1909  }%

  Finally the all caps version:

1910    \ifcsdef{@GLS@user@#1@}%
1911    {%
1912      \PackageError{glossaries}%
1913      {Can't define '\string#7' as helper command
1914      '\expandafter\string\csname @GLS@user@#1@\endcsname' already exists}%
1915    }%
1916  }%
1917  {%

1918    \newrobustcmd*{#7}{\@gls@hyp@opt{\csuse{@GLS@user@#1}}}%
1919    \expandafter\newcommand\expandafter*\expandafter
1920    {\csname @GLS@user@#1\endcsname}[2][{}]{%
1921      \new@ifnextchar[%
1922        {\csuse{@GLS@user@#1@}{##1}{##2}}%
1923        {\csuse{@GLS@user@#1@}{##1}{##2}[]}}%
1924    \csdef{@GLS@user@#1@}##1##2[##3]{%
1925      \@gls@field@link{##1}{##2}{\mfirstucMakeUppercase{#3{##2}##3}}%
1926    }%
1927  }%
1928 }%
1929 {%
1930   \PackageError{glossaries}{Key '#1' already exists}{}%
1931 }%
1932 }

```

\gls@writedefhook

```

1933 \newcommand*{\gls@writedefhook}{}

```

\gls@assign@desc

```

1934 \newcommand*{\gls@assign@desc}[1]{%
1935   \gls@assign@field{#1}{desc}{\@glo@desc}%
1936   \gls@assign@field{\@glo@desc}{#1}{descplural}{\@glo@descplural}%
1937 }

```

ongnewglossaryentry

```

1938 \newcommand{\longnewglossaryentry}[3]{%
1939   \glsdoifnoexists{#1}%
1940   {%
1941     \bgroup
1942     \let\@org@newglossaryentryprehook\@newglossaryentryprehook
1943     \long\def\@newglossaryentryprehook{%
1944       \long\def\@glo@desc{#3\leavevmode\unskip\nopostdesc}%
1945       \@org@newglossaryentryprehook
1946     }%

```

```

1947     \renewcommand*\gls@assign@desc}[1]{%
1948         \global\cslet{glo@glsdetoklabel{#1}@desc}{\@glo@desc}%
1949         \global\cslet{glo@glsdetoklabel{#1}@descplural}{\@glo@desc}%
1950     }
1951     \gls@defglossaryentry{#1}{#2}%
1952 \egroup
1953 }
1954 }

```

Only allowed in the preamble. (Otherwise a long description could cause problems when writing the entry definition to the temporary file.)

```
1955 \@onlypreamble{\longnewglossaryentry}
```

`\longprovideglossaryentry` As the above but only defines the entry if it doesn't already exist.

```

1956 \newcommand{\longprovideglossaryentry}[3]{%
1957     \ifglentryexists{#1}{}%
1958     {\longnewglossaryentry{#1}{#2}{#3}}%
1959 }
1960 \@onlypreamble{\longprovideglossaryentry}

```

`\gls@defglossaryentry` `\gls@defglossaryentry{<label>}{<key-val list>}`

Defines a new entry without checking if it already exists.

```
1961 \newcommand{\gls@defglossaryentry}[2]{%
```

Store label

```
1962     \edef\@glo@label{\glsdetoklabel{#1}}%
```

Provide a means for user defined keys to reference the label:

```
1963     \let\glslabel\@glo@label
```

Set up defaults. If the name or description keys are omitted, an error will be generated.

```
1964     \let\@glo@name\@glsnoname
```

```
1965     \let\@glo@desc\@glsnodesc
```

```
1966     \let\@glo@descplural\@gls@default@value
```

```
1967     \let\@glo@type\@gls@default@value
```

```
1968     \let\@glo@symbol\@gls@default@value
```

```
1969     \let\@glo@symbolplural\@gls@default@value
```

```
1970     \let\@glo@text\@gls@default@value
```

```
1971     \let\@glo@plural\@gls@default@value
```

Using `\let` instead of `\def` to make later comparison avoid expansion issues.

(Thanks to Ulrich Diez for suggesting this.)

```
1972     \let\@glo@first\@gls@default@value
```

1973 \let\@glo@firstplural\@gls@default@value

Set the default sort:

1974 \let\@glo@sort\@gls@default@value

Set the default counter:

1975 \let\@glo@counter\@gls@default@value

1976 \def\@glo@see{}%

1977 \def\@glo@parent{}%

1978 \def\@glo@prefix{}%

1979 \def\@glo@useri{}%

1980 \def\@glo@userii{}%

1981 \def\@glo@useriii{}%

1982 \def\@glo@useriv{}%

1983 \def\@glo@userv{}%

1984 \def\@glo@uservi{}%

1985 \def\@glo@short{}%

1986 \def\@glo@shortpl{}%

1987 \def\@glo@long{}%

1988 \def\@glo@longpl{}%

Add start hook in case another package wants to add extra keys.

1989 \@newglossaryentryprehook

Extract key-val information from third parameter:

1990 \setkeys{glossentry}{#2}%

Check there is a default glossary.

1991 \ifundef\glsdefaulttype

1992 {%

1993 \PackageError{glossaries}%

1994 {No default glossary type (have you used ‘nomain’?)}%

1995 {If you use package option ‘nomain’ you must define

1996 a new glossary before you can define entries}%

1997 }%

1998 }%

Assign type. This must be fully expandable

1999 \gls@assign@field{\glsdefaulttype}{\@glo@label}{type}{\@glo@type}%

2000 \edef\@glo@type{\glsentrytype{\@glo@label}}%

Check to see if this glossary type has been defined, if it has, add this label to the relevant list, otherwise generate an error.

2001 \ifcsundef{glolist@\@glo@type}%

2002 {%

2003 \PackageError{glossaries}%

2004 {Glossary type ‘\@glo@type’ has not been defined}%

```

2005      {You need to define a new glossary type, before making entries
2006      in it}%
2007    }%
2008    {%

```

Check if it's an ignored glossary

```

2009      \ifignoredglossary\@glo@type
2010    {%

```

The description may be omitted for an entry in an ignored glossary.

```

2011      \ifx\@glo@desc\@glsnodesc
2012        \let\@glo@desc\@empty
2013      \fi
2014    }%
2015    {%
2016    }%
2017    \protected@edef\@glo@list@{\csname glo@list@\@glo@type\endcsname}%
2018    \expandafter\xdef\csname glo@list@\@glo@type\endcsname{%
2019      \@glo@list@{\@glo@label},}%
2020    }%

```

Initialise level to 0.

```

2021    \gls@level=0\relax

```

Has this entry been assigned a parent?

```

2022    \ifx\@glo@parent\@empty

```

Doesn't have a parent. Set \glo@<label>@parent to empty.

```

2023      \expandafter\gdef\csname glo@\@glo@label @parent\endcsname{}%
2024    \else

```

Has a parent. Check to ensure this entry isn't its own parent.

```

2025      \ifdefequal\@glo@label\@glo@parent%
2026    {%
2027      \PackageError{glossaries}{Entry '@@glo@label' can't be its own parent}{}%
2028      \def\@glo@parent{}%
2029      \expandafter\gdef\csname glo@\@glo@label @parent\endcsname{}%
2030    }%
2031    {%

```

Check the parent exists:

```

2032      \ifglentryexists{\@glo@parent}%
2033    {%

```

Parent exists. Set \glo@<label>@parent.

```

2034      \expandafter\xdef\csname glo@\@glo@label @parent\endcsname{%
2035        \@glo@parent}%

```

Determine level.

```

2036      \gls@level=\csname glo@\@glo@parent @level\endcsname\relax
2037      \advance\gls@level by 1\relax

```


If name hasn't been specified, use same as the parent name

```

2038      \ifx\@glo@name\@glsnoname
2039      \expandafter\let\expandafter\@glo@name
2040      \csname glo@\@glo@parent @name\endcsname

```

If name and plural haven't been specified, use same as the parent

```

2041      \ifx\@glo@plural\@gls@default@value
2042      \expandafter\let\expandafter\@glo@plural
2043      \csname glo@\@glo@parent @plural\endcsname
2044      \fi
2045      \fi
2046      }%
2047      {%

```

Parent doesn't exist, so issue an error message and change this entry to have no parent

```

2048      \PackageError{glossaries}%
2049      {%
2050      Invalid parent '\@glo@parent'
2051      for entry '\@glo@label' - parent doesn't exist%
2052      }%
2053      {%
2054      Parent entries must be defined before their children%
2055      }%
2056      \def\@glo@parent{%
2057      \expandafter\gdef\csname glo@\@glo@label @parent\endcsname{%
2058      }%
2059      }%
2060      \fi

```

Set the level for this entry

```

2061      \expandafter\xdef\csname glo@\@glo@label @level\endcsname{\number\@gls@level}%

```

Define commands associated with this entry:

```

2062      \gls@assign@field{\@glo@name}{\@glo@label}{sortvalue}{\@glo@sort}%
2063      \letcs\@glo@sort{glo@\@glo@label @sortvalue}%
2064      \gls@assign@field{\@glo@name}{\@glo@label}{text}{\@glo@text}%
2065      \expandafter\gls@assign@field\expandafter
2066      {\csname glo@\@glo@label @text\endcsname\glspluralsuffix}%
2067      {\@glo@label}{plural}{\@glo@plural}%
2068      \expandafter\gls@assign@field\expandafter
2069      {\csname glo@\@glo@label @text\endcsname}%
2070      {\@glo@label}{first}{\@glo@first}%

```

If first has been specified, make the default by appending \glspluralsuffix, otherwise make the default the value of the plural key.

```

2071      \ifx\@glo@first\@gls@default@value
2072      \expandafter\gls@assign@field\expandafter
2073      {\csname glo@\@glo@label @plural\endcsname}%
2074      {\@glo@label}{firstpl}{\@glo@firstplural}%
2075      \else

```

```

2076 \expandafter\gls@assign@field\expandafter
2077 {\csname glo@\@glo@label @first\endcsname\glspluralsuffix}%
2078 {\@glo@label}{firstpl}{\@glo@firstplural}%
2079 \fi

2080 \ifcsundef{@glo@type@\@glo@type @counter}%
2081 {%
2082 \def\@glo@defaultcounter{\glscounter}%
2083 }%
2084 {%
2085 \letcs\@glo@defaultcounter{@glo@type@\@glo@type @counter}%
2086 }%
2087 \gls@assign@field{\@glo@defaultcounter}{\@glo@label}{counter}{\@glo@counter}%
2088 \gls@assign@field{}{\@glo@label}{useri}{\@glo@useri}%
2089 \gls@assign@field{}{\@glo@label}{userii}{\@glo@userii}%
2090 \gls@assign@field{}{\@glo@label}{useriii}{\@glo@useriii}%
2091 \gls@assign@field{}{\@glo@label}{useriv}{\@glo@useriv}%
2092 \gls@assign@field{}{\@glo@label}{userv}{\@glo@userv}%
2093 \gls@assign@field{}{\@glo@label}{uservi}{\@glo@uservi}%
2094 \gls@assign@field{}{\@glo@label}{short}{\@glo@short}%
2095 \gls@assign@field{}{\@glo@label}{shortpl}{\@glo@shortpl}%
2096 \gls@assign@field{}{\@glo@label}{long}{\@glo@long}%
2097 \gls@assign@field{}{\@glo@label}{longpl}{\@glo@longpl}%
2098 \ifx\@glo@name\@glsnoname
2099 \@glsnoname
2100 \let\@glo@name\@gls@default@value
2101 \fi
2102 \gls@assign@field{}{\@glo@label}{name}{\@glo@name}%

```

Set default numberlist if not defined:

```

2103 \ifcsundef{glo@\@glo@label @numberlist}%
2104 {%
2105 \csxdef{glo@\@glo@label @numberlist}{%
2106 \noexpand\@gls@missingnumberlist{\@glo@label}}%
2107 }%
2108 {}%

```

The smaller and smallcaps options set the description to \@glo@first. Need to check for this, otherwise it won't get expanded if the description gets sanitized.

```

2109 \def\@glo@@desc{\@glo@first}%
2110 \ifx\@glo@desc\@glo@@desc
2111 \let\@glo@desc\@glo@first
2112 \fi
2113 \ifx\@glo@desc\@glsnodesc
2114 \@glsnodesc
2115 \let\@glo@desc\@gls@default@value
2116 \fi
2117 \gls@assign@desc{\@glo@label}%

```

Set the sort key for this entry:

```

2118 \gls@defsort{\@glo@type}{\@glo@label}%
2119 \def\@glo@@symbol{\@glo@text}%
2120 \ifx\@glo@symbol\@glo@@symbol
2121 \let\@glo@symbol\@glo@text
2122 \fi
2123 \gls@assign@field{\relax}{\@glo@label}{symbol}{\@glo@symbol}%
2124 \expandafter
2125 \gls@assign@field\expandafter
2126 {\csname glo@\@glo@label @symbol\endcsname}
2127 {\@glo@label}{symbolplural}{\@glo@symbolplural}%

```

Define an associated boolean variable to determine whether this entry has been used yet (needs to be defined globally):

```

2128 \expandafter\xdef\csname glo@\@glo@label @flagfalse\endcsname{%
2129 \noexpand\global
2130 \noexpand\let\expandafter\noexpand
2131 \csname ifglo@\@glo@label @flag\endcsname\noexpand\iffalse
2132 }%
2133 \expandafter\xdef\csname glo@\@glo@label @flagtrue\endcsname{%
2134 \noexpand\global
2135 \noexpand\let\expandafter\noexpand
2136 \csname ifglo@\@glo@label @flag\endcsname\noexpand\iftrue
2137 }%
2138 \csname glo@\@glo@label @flagfalse\endcsname

```

Sort out any cross-referencing if required.

```

2139 \ifdefined\@glo@see
2140 {}%
2141 {%
2142 \protected@edef\@do@glsssee{%
2143 \noexpand\gls@fixbraces\noexpand\@glo@list\@glo@see
2144 \noexpand\@nil
2145 \noexpand\expandafter\noexpand\@glsssee\noexpand\@glo@list{\@glo@label}}%
2146 \@do@glsssee
2147 }%

```

Determine and store main part of the entry's index format.

```

2148 \ifignoredglossary\@glo@type
2149 {%
2150 \csdef{glo@\@glo@label @index}{}%
2151 }
2152 {%
2153 \do@glo@storeentry{\@glo@label}%
2154 }%

```

Add end hook in case another package wants to add extra keys.

```

2155 \@newglossaryentryposthook
2156 }

```

`glossaryentryprehook` Allow extra information to be added to glossary entries:

```
2157 \newcommand*{\@newglossaryentryprehook}{}
```

`glossaryentryposthook` Allow extra information to be added to glossary entries:

```
2158 \newcommand*{\@newglossaryentryposthook}{}
```

`\glsmoveentry` Moves entry whose label is given by first argument to the glossary named in the second argument.

```
2159 \newcommand*{\glsmoveentry}[2]{%
2160   \edef\@glo@thislabel{\glstetoklabel{#1}}%
2161   \edef\glo@type{\csname glo@\@glo@thislabel @type\endcsname}%
2162   \def\glo@list{,}%
2163   \forlslentries[\glo@type]{\glo@label}%
2164   {%
2165     \ifdefequal\@glo@thislabel\glo@label
2166     {\eappto\glo@list{\glo@label,}}%
2167   }%
2168   \cslet\glo@list@\glo@type{\glo@list}%
2169   \csdef{glo@\@glo@thislabel @type}{#2}%
2170 }
```

`@glossaryentryfield` Indicate what command should be used to display each entry in the glossary. (This enables the glossaries-accsupp package to use `\accsuppglossaryentryfield` instead.)

```
2171 \ifglxindy
2172   \newcommand*{\@glossaryentryfield}{\string\glossentry}
2173 \else
2174   \newcommand*{\@glossaryentryfield}{\string\glossentry}
2175 \fi
```

`glossarysubentryfield` Indicate what command should be used to display each subentry in the glossary. (This enables the glossaries-accsupp package to use `\accsuppglossarysubentryfield` instead.)

```
2176 \ifglxindy
2177   \newcommand*{\@glossarysubentryfield}{%
2178     \string\subglossentry}
2179 \else
2180   \newcommand*{\@glossarysubentryfield}{%
2181     \string\subglossentry}
2182 \fi
```

`\@glo@storeentry` `\@glo@storeentry{<label>}`

Determine the format to write the entry in the glossary output (`.glo`) file. The argument is the entry's label (should already have been de-tok'ed if required).

The result is stored in `\glo@<label>@index`, where `<label>` is the entry's label.
(This doesn't include any formatting or location information.)

```
2183 \newcommand{\@glo@storeentry}[1]{%
```

Escape makeindex/xindy special characters in the label:

```
2184 \edef\@glo@esclabel{#1}%
2185 \@gls@checkmkidxchars\@glo@esclabel
```

Get the sort string and escape any special characters

```
2186 \protected@edef\@glo@sort{\csname glo@#1@sort\endcsname}%
2187 \@gls@checkmkidxchars\@glo@sort
```

Same again for the name string. Escape any special characters in the prefix

```
2188 \@gls@checkmkidxchars\@glo@prefix
```

Get the parent, if one exists

```
2189 \edef\@glo@parent{\csname glo@#1@parent\endcsname}%
```

Write the information to the glossary file.

```
2190 \ifglsxindy
```

Store using xindy syntax.

```
2191 \ifx\@glo@parent\@empty
```

Entry doesn't have a parent

```
2192 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
2193 (\string"\@glo@sort\string" %
2194 \string"\@glo@prefix\@glossaryentryfield{\@glo@esclabel}\string") %
2195 }%
2196 \else
```

Entry has a parent

```
2197 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
2198 \csname glo@\@glo@parent @index\endcsname
2199 (\string"\@glo@sort\string" %
2200 \string"\@glo@prefix\@glossarysubentryfield
2201 {\csname glo@#1@level\endcsname}{\@glo@esclabel}\string") %
2202 }%
2203 \fi
2204 \else
```

Store using makeindex syntax.

```
2205 \ifx\@glo@parent\@empty
```

Sanitize `\@glo@prefix`

```
2206 \@onelevel@sanitize\@glo@prefix
```

Entry doesn't have a parent

```
2207 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
2208 \@glo@sort\@gls@actualchar\@glo@prefix
2209 \@glossaryentryfield{\@glo@esclabel}%
2210 }%
2211 \else
```

Entry has a parent

```

2212 \expandafter\protected\edef\csname glo@#1@index\endcsname{%
2213 \csname glo@\@glo@parent @index\endcsname\@gls@levelchar
2214 \@glo@sort\@gls@actualchar\@glo@prefix
2215 \@glossarysubentryfield
2216 {\csname glo@#1@level\endcsname}\@glo@esclabel}%
2217 }%
2218 \fi
2219 \fi
2220 }

```

1.8 Resetting and unsetting entry flags

Each glossary entry is assigned a conditional of the form `\ifglo@<label>@flag` which determines whether or not the entry has been used (see also `\ifglsused` defined below). These flags can be set and unset using the following macros, but first we need to know if we're in `amsmath`'s `align` environment's measuring pass.

`\gls@ifnotmeasuring`

```

2221 \AtBeginDocument{%
2222 \ifpackageloaded{amsmath}%
2223 {\let\gls@ifnotmeasuring\@gls@ifnotmeasuring}%
2224 }%
2225 }
2226 \newcommand*{\@gls@ifnotmeasuring}[1]{%
2227 \ifmeasuring@
2228 \else
2229 #1%
2230 \fi
2231 }
2232 \newcommand*\gls@ifnotmeasuring[1]{#1}

```

`\glsreset` The command `\glsreset{<label>}` can be used to set the entry flag to indicate that it hasn't been used yet. The required argument is the entry label.

```

2233 \newcommand*\glsreset[1]{%
2234 \gls@ifnotmeasuring
2235 {%
2236 \glsdoifexists{#1}%
2237 {%
2238 \expandafter\global\csname glo@\glsdetoklabel{#1}@flagfalse\endcsname
2239 }%
2240 }%
2241 }

```

`\glslocalreset` As above, but with only a local effect:

```

2242 \newcommand*\glslocalreset[1]{%
2243 \gls@ifnotmeasuring

```

```

2244  {%
2245    \glsdoifexists{#1}%
2246    {%
2247      \expandafter\let\csname ifglo@\glsdetoklabel{#1}@flag\endcsname\iffalse
2248    }%
2249  }%
2250 }

```

`\glsunset` The command `\glsunset{<label>}` can be used to set the entry flag to indicate that it has been used. The required argument is the entry label.

```

2251 \newcommand*{\glsunset}[1]{%
2252   \gls@ifnotmeasuring
2253   {%
2254     \glsdoifexists{#1}%
2255     {%
2256       \expandafter\global\csname glo@\glsdetoklabel{#1}@flagtrue\endcsname
2257     }%
2258   }%
2259 }

```

`\glslocalunset` As above, but with only a local effect:

```

2260 \newcommand*{\glslocalunset}[1]{%
2261   \gls@ifnotmeasuring
2262   {%
2263     \glsdoifexists{#1}%
2264     {%
2265       \expandafter\let\csname ifglo@\glsdetoklabel{#1}@flag\endcsname\iftrue
2266     }%
2267   }%
2268 }

```

Reset all entries for the named glossaries (supplied in a comma-separated list).

Syntax: `\glsresetall[<glossary-list>]`

`\glsresetall`

```

2269 \newcommand*{\glsresetall}[1][\@glo@types]{%
2270   \forallglsentries[#1]{\@glsentry}%
2271   {%
2272     \glsreset{\@glsentry}%
2273   }%
2274 }

```

As above, but with only a local effect:

`\glslocalresetall`

```

2275 \newcommand*{\glslocalresetall}[1][\@glo@types]{%
2276   \forallglsentries[#1]{\@glsentry}%
2277   {%
2278     \glslocalreset{\@glsentry}%

```

```
2279 }%
2280 }
```

Unset all entries for the named glossaries (supplied in a comma-separated list).
 Syntax: `\glsunsetall[<glossary-list>]`

`\glsunsetall`

```
2281 \newcommand*{\glsunsetall}[1][\@gls@types]{%
2282   \forallglsentries[#1]{\@glsentry}%
2283   {%
2284     \glsunset{\@glsentry}%
2285   }%
2286 }
```

As above, but with only a local effect:

`\glslocalunsetall`

```
2287 \newcommand*{\glslocalunsetall}[1][\@gls@types]{%
2288   \forallglsentries[#1]{\@glsentry}%
2289   {%
2290     \glslocalunset{\@glsentry}%
2291   }%
2292 }
```

1.9 Loading files containing glossary entries

Glossary entries can be defined in an external file. These external files can contain `\newglossaryentry` and `\newacronym` commands.¹

`\loadglsentries[<type>]{<filename>}`

This command will input the file using `\input`. The optional argument specifies to which glossary the entries should be assigned if they haven't used the type key. If the optional argument is not specified, the default glossary is used. Only those entries used in the document (via `\glslink`, `\gls`, `\glspl` and uppercase variants or `\glsadd` and `\glsaddall` will appear in the glossary). The mandatory argument is the filename (with or without `.tex` extension).

`\loadglsentries`

```
2293 \newcommand*{\loadglsentries}[2][\@gls@default]{%
2294   \let\@gls@default\glsdefaulttype
2295   \def\glsdefaulttype{#1}\input{#2}%
2296   \let\glsdefaulttype\@gls@default
2297 }
```

`\loadglsentries` can only be used in the preamble:

```
2298 \@onlypreamble{\loadglsentries}
```

¹and any other valid \LaTeX code that can be used in the preamble.

1.10 Using glossary entries in the text

Any term that has been defined using `\newglossaryentry` (or `\newacronym`) can be displayed in the text (i.e. outside of the glossary) using one of the commands defined in this section. Unless you use `\glslink`, the way the term appears in the text is determined by `\glsdisplayfirst` (if it is the first time the term has been used) or `\glsdisplay` (for subsequent use). Any formatting commands (such as `\textbf`) is governed by `\glstextformat`. By default this just displays the link text “as is”.

`\glstextformat`

```
2299 \newcommand*{\glstextformat}[1]{#1}
```

`\glsentryfmt` As from version 3.11a, the way in which an entry is displayed is now governed by `\glsentryfmt`. This doesn't take any arguments. The required information is set by commands like `\gls`. To ensure backward compatibility, the default use the old `\glsdisplay` and `\glsdisplayfirst` style of commands

```
2300 \newcommand*{\glsentryfmt}{%
```

```
2301   \@@gls@default@entryfmt\glsdisplayfirst\glsdisplay
```

```
2302 }
```

Format that provides backwards compatibility:

```
2303 \newcommand*{\@@gls@default@entryfmt}[2]{%
```

```
2304   \ifdefempty\glscustomtext
```

```
2305   {%
```

```
2306     \glsifplural
```

```
2307     {%
```

Plural form

```
2308     \glscapscase
```

```
2309     {%
```

Don't adjust case

```
2310     \ifglsused\glslabel
```

```
2311     {%
```

Subsequent use

```
2312     #2{\glsentryplural{\glslabel}}%
```

```
2313     {\glsentrydescplural{\glslabel}}%
```

```
2314     {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
```

```
2315     }%
```

```
2316     {%
```

First use

```
2317     #1{\glsentryfirstplural{\glslabel}}%
```

```
2318     {\glsentrydescplural{\glslabel}}%
```

```
2319     {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
```

```
2320     }%
```

```
2321     }%
```

```
2322     {%
```

Make first letter upper case

```
2323      \ifglused\glslabel
2324      {%
```

Subsequent use. (Expansion was used in version 3.07 and below in case the name wasn't the first thing to be displayed, but now the user can sort out the upper casing in `\defglentryfmt`, which avoids the issues caused by fragile commands.)

```
2325      \ifbool{glscompatible-3.07}%
2326      {%
2327      \protected@edef\@glo@etext{%
2328      #2{\glentryplural{\glslabel}}%
2329      {\glentrydescplural{\glslabel}}%
2330      {\glentrysymbolplural{\glslabel}}{\glsinsert}}%
2331      \xmakefirstuc\@glo@etext
2332      }%
2333      {%
2334      #2{\Glsentryplural{\glslabel}}%
2335      {\glentrydescplural{\glslabel}}%
2336      {\glentrysymbolplural{\glslabel}}{\glsinsert}%
2337      }%
2338      }%
2339      {%
```

First use

```
2340      \ifbool{glscompatible-3.07}%
2341      {%
2342      \protected@edef\@glo@etext{%
2343      #1{\glentryfirstplural{\glslabel}}%
2344      {\glentrydescplural{\glslabel}}%
2345      {\glentrysymbolplural{\glslabel}}{\glsinsert}}%
2346      \xmakefirstuc\@glo@etext
2347      }%
2348      {%
2349      #1{\Glsentryfirstplural{\glslabel}}%
2350      {\glentrydescplural{\glslabel}}%
2351      {\glentrysymbolplural{\glslabel}}{\glsinsert}%
2352      }%
2353      }%
2354      }%
2355      {%
```

Make all upper case

```
2356      \ifglused\glslabel
2357      {%
```

Subsequent use

```
2358      \mfirstucMakeUppercase{#2{\glentryplural{\glslabel}}%
2359      {\glentrydescplural{\glslabel}}%
2360      {\glentrysymbolplural{\glslabel}}{\glsinsert}}%
```

```

2361         }%
2362         {%

First use
2363         \mfirstucMakeUppercase{#1{\glsentryfirstplural{\glslabel}}}%
2364         {\glsentrydescplural{\glslabel}}}%
2365         {\glsentrysymbolplural{\glslabel}}{\glsinsert}}}%
2366     }%
2367 }%
2368 }%
2369 {%

Singular form
2370     \glscapscase
2371     {%

Don't adjust case
2372     \ifglused\glslabel
2373     {%

Subsequent use
2374     #2{\glsentrytext{\glslabel}}}%
2375     {\glsentrydesc{\glslabel}}}%
2376     {\glsentrysymbol{\glslabel}}{\glsinsert}}}%
2377 }%
2378 {%

First use
2379     #1{\glsentryfirst{\glslabel}}}%
2380     {\glsentrydesc{\glslabel}}}%
2381     {\glsentrysymbol{\glslabel}}{\glsinsert}}}%
2382 }%
2383 }%
2384 {%

Make first letter upper case
2385     \ifglused\glslabel
2386     {%

Subsequent use
2387     \ifbool{glsc-compatible-3.07}%
2388     {%
2389         \protected@edef\@glo@etext{%
2390             #2{\glsentrytext{\glslabel}}}%
2391             {\glsentrydesc{\glslabel}}}%
2392             {\glsentrysymbol{\glslabel}}{\glsinsert}}}%
2393         \xmakefirstuc\@glo@etext
2394     }%
2395     {%
2396         #2{\Glsentrytext{\glslabel}}}%
2397         {\Glsentrydesc{\glslabel}}}%
2398         {\Glsentrysymbol{\glslabel}}{\glsinsert}}}%

```

```

2399         }%
2400     }%
2401     {%

    First use
2402         \ifbool{glscompatible-3.07}%
2403     {%
2404         \protected@edef\@glo@etext{%
2405             #1{\glsentryfirst{\glslabel}}%
2406             {\glsentrydesc{\glslabel}}%
2407             {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2408         \xmakefirstuc\@glo@etext
2409     }%
2410     {%
2411         #1{\Glsentryfirst{\glslabel}}%
2412         {\glsentrydesc{\glslabel}}%
2413         {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2414     }%
2415 }%
2416 }%
2417 {%

    Make all upper case
2418     \ifglsused\glslabel
2419     {%

    Subsequent use
2420         \mfirstucMakeUppercase{#2{\glsentrytext{\glslabel}}%
2421         {\glsentrydesc{\glslabel}}%
2422         {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2423     }%
2424     {%

    First use
2425         \mfirstucMakeUppercase{#1{\glsentryfirst{\glslabel}}%
2426         {\glsentrydesc{\glslabel}}%
2427         {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2428     }%
2429 }%
2430 }%
2431 }%
2432 {%

    Custom text provided in \glsdisp
2433     \ifglsused{\glslabel}%
2434     {%

    Subsequent use
2435         #2{\glscustomtext}%
2436         {\glsentrydesc{\glslabel}}%
2437         {\glsentrysymbol{\glslabel}}{%
2438     }%

```

```

2439      {%
      First use
2440      #1{\glscustomtext}%
2441      {\glentrydesc{\glslabel}}%
2442      {\glentrysymbol{\glslabel}}}%
2443      }%
2444      }%
2445      }

```

`\glsgenentryfmt` Define a generic format that just uses the first, text, plural or first plural keys (or the custom text) with the insert text appended.

```

2446 \newcommand*{\glsgenentryfmt}{%
2447   \ifdefempty\glscustomtext
2448   {%
2449     \glsifplural
2450     {%

```

Plural form

```

2451     \glscapscase
2452     {%

```

Don't adjust case

```

2453     \ifglsused\glslabel
2454     {%

```

Subsequent use

```

2455     \glentryplural{\glslabel}\glsinsert
2456     }%
2457     }%

```

First use

```

2458     \glentryfirstplural{\glslabel}\glsinsert
2459     }%
2460     }%
2461     {%

```

Make first letter upper case

```

2462     \ifglsused\glslabel
2463     {%

```

Subsequent use.

```

2464     \Glsentryplural{\glslabel}\glsinsert
2465     }%
2466     }%

```

First use

```

2467     \Glsentryfirstplural{\glslabel}\glsinsert
2468     }%
2469     }%
2470     {%

```

Make all upper case

```
2471      \ifglsused\glslabel
2472      {%
```

Subsequent use

```
2473      \mfirstucMakeUppercase
2474      {\glsentryplural{\glslabel}\glsinsert}%
2475      }%
2476      {%
```

First use

```
2477      \mfirstucMakeUppercase
2478      {\glsentryfirstplural{\glslabel}\glsinsert}%
2479      }%
2480      }%
2481      }%
2482      {%
```

Singular form

```
2483      \glscapscase
2484      {%
```

Don't adjust case

```
2485      \ifglsused\glslabel
2486      {%
```

Subsequent use

```
2487      \glsentrytext{\glslabel}\glsinsert
2488      }%
2489      {%
```

First use

```
2490      \glsentryfirst{\glslabel}\glsinsert
2491      }%
2492      }%
2493      {%
```

Make first letter upper case

```
2494      \ifglsused\glslabel
2495      {%
```

Subsequent use

```
2496      \Glsentrytext{\glslabel}\glsinsert
2497      }%
2498      {%
```

First use

```
2499      \Glsentryfirst{\glslabel}\glsinsert
2500      }%
2501      }%
2502      {%
```

Make all upper case

```
2503      \ifglsused\glslabel
2504      {%
```

Subsequent use

```
2505      \mfirstucMakeUppercase{\glsentrytext{\glslabel}\glsinsert}%
2506      }%
2507      {%
```

First use

```
2508      \mfirstucMakeUppercase{\glsentryfirst{\glslabel}\glsinsert}%
2509      }%
2510      }%
2511      }%
2512      }%
2513      {%
```

Custom text provided in \glsdisp. (The insert is most likely to be empty at this point.)

```
2514      \glscustomtext\glsinsert
2515      }%
2516 }
```

`\glsngenacfmt` Define a generic acronym format that uses the long and short keys (or their plurals) and `\acrfullformat`, `\firstacronymfont` and `\acronymfont`.

```
2517 \newcommand*{\glsngenacfmt}{%
2518   \ifdefempty\glscustomtext
2519   {%
2520     \ifglsused\glslabel
2521     {%
```

Subsequent use:

```
2522     \glsifplural
2523     {%
```

Subsequent plural form:

```
2524     \glscapscase
2525     {%
```

Subsequent plural form, don't adjust case:

```
2526     \acronymfont{\glsentryshortpl{\glslabel}}\glsinsert
2527     }%
2528     {%
```

Subsequent plural form, make first letter upper case:

```
2529     \acronymfont{\Glsentryshortpl{\glslabel}}\glsinsert
2530     }%
2531     {%
```

Subsequent plural form, all caps:

```
2532     \mfirstucMakeUppercase
2533     {\acronymfont{\glsentryshortpl{\glslabel}}\glsinsert}%
```

2534 }%
 2535 }%
 2536 {%

Subsequent singular form

2537 \glscapscase
 2538 {%

Subsequent singular form, don't adjust case:

2539 \acronymfont{\glstryshort{\glslabel}}\glsinsert
 2540 }%
 2541 {%

Subsequent singular form, make first letter upper case:

2542 \acronymfont{\Glsstryshort{\glslabel}}\glsinsert
 2543 }%
 2544 {%

Subsequent singular form, all caps:

2545 \mfirstucMakeUppercase
 2546 {\acronymfont{\glstryshort{\glslabel}}\glsinsert}%
 2547 }%
 2548 }%
 2549 }%
 2550 {%

First use:

2551 \glsifplural
 2552 {%

First use plural form:

2553 \glscapscase
 2554 {%

First use plural form, don't adjust case:

2555 \genplacrfullformat{\glslabel}{\glsinsert}%
 2556 }%
 2557 {%

First use plural form, make first letter upper case:

2558 \Genplacrfullformat{\glslabel}{\glsinsert}%
 2559 }%
 2560 {%

First use plural form, all caps:

2561 \mfirstucMakeUppercase
 2562 {\genplacrfullformat{\glslabel}{\glsinsert}}%
 2563 }%
 2564 }%
 2565 {%

First use singular form

2566 \glscapscase
 2567 {%

First use singular form, don't adjust case:

```
2568      \genacrfullformat{\glslabel}{\glsinsert}%
2569      }%
2570      {%
```

First use singular form, make first letter upper case:

```
2571      \Genacrfullformat{\glslabel}{\glsinsert}%
2572      }%
2573      {%
```

First use singular form, all caps:

```
2574      \mfirstucMakeUppercase
2575      {\genacrfullformat{\glslabel}{\glsinsert}}%
2576      }%
2577      }%
2578      }%
2579      }%
2580      {%
```

User supplied text.

```
2581      \glscustomtext
2582      }%
2583 }
```

`\genacrfullformat` `\genacrfullformat{<label>}{<insert>}`

The full format used by `\glsacrfmt` (singular).

```
2584 \newcommand*{\genacrfullformat}[2]{%
2585   \glsentrylong{#1}#2\space
2586   (\protect\firstacronymfont{\glsentryshort{#1}})%
2587 }
```

`\Genacrfullformat` `\Genacrfullformat{<label>}{<insert>}`

As above but makes the first letter upper case.

```
2588 \newcommand*{\Genacrfullformat}[2]{%
2589   \protected@edef\gls@text{\genacrfullformat{#1}{#2}}%
2590   \xmakefirstuc\gls@text
2591 }
```

`\genplacrfullformat` `\genplacrfullformat{<label>}{<insert>}`

The full format used by `\glsacrfmt` (plural).

```
2592 \newcommand*{\genplacrfullformat}[2]{%
2593   \glsentrylongpl{#1}#2\space
2594   (\protect\firstacronymfont{\glsentryshortpl{#1}})%
```

2595 }

`\Genplacrfullformat` `\Genplacrfullformat{<label>}{<insert>}`

As above but makes the first letter upper case.

```
2596 \newcommand*{\Genplacrfullformat}[2]{%
2597   \protected@edef\gls@text{\genplacrfullformat{#1}{#2}}%
2598   \xmakefirstuc\gls@text
2599 }
```

`\glsdisplayfirst` Deprecated. Kept for backward compatibility.

```
2600 \newcommand*{\glsdisplayfirst}[4]{#1#4}
```

`\glsdisplay` Deprecated. Kept for backward compatibility.

```
2601 \newcommand*{\glsdisplay}[4]{#1#4}
```

`\defglsdisplay` Deprecated. Kept for backward compatibility.

```
2602 \newcommand*{\defglsdisplay}[2][\glsdefaulttype]{%
2603   \GlossariesWarning{\string\defglsdisplay\space is now obsolete.^^J
2604   Use \string\defglsentryfmt\space instead}%
2605   \expandafter\def\csname gls@#1@display\endcsname##1##2##3##4{#2}%
2606   \edef\@gls@doentrydef{%
2607     \noexpand\defglsentryfmt[#1]{%
2608       \noexpand\ifcsdef{gls@#1@displayfirst}%
2609       {%
2610         \noexpand\@gls@default@entryfmt
2611         {\noexpand\csuse{gls@#1@displayfirst}}%
2612         {\noexpand\csuse{gls@#1@display}}%
2613       }%
2614       {%
2615         \noexpand\@gls@default@entryfmt
2616         {\noexpand\glsdisplayfirst}%
2617         {\noexpand\csuse{gls@#1@display}}%
2618       }%
2619     }%
2620   }%
2621   \@gls@doentrydef
2622 }
```

`\defglsdisplayfirst` Deprecated. Kept for backward compatibility.

```
2623 \newcommand*{\defglsdisplayfirst}[2][\glsdefaulttype]{%
2624   \GlossariesWarning{\string\defglsdisplayfirst\space is now obsolete.^^J
2625   Use \string\defglsentryfmt\space instead}%
2626   \expandafter\def\csname gls@#1@displayfirst\endcsname##1##2##3##4{#2}%
2627   \edef\@gls@doentrydef{%
2628     \noexpand\defglsentryfmt[#1]{%
2629       \noexpand\ifcsdef{gls@#1@display}%

```

```

2630      {%
2631      \noexpand\@gls@default@entryfmt
2632      {\noexpand\csuse{gls@#1@displayfirst}}}%
2633      {\noexpand\csuse{gls@#1@display}}}%
2634      }%
2635      {%

2636      \noexpand\@gls@default@entryfmt
2637      {\noexpand\csuse{gls@#1@displayfirst}}}%
2638      {\noexpand\glsdisplay}%
2639      }%
2640      }%
2641      }%
2642      \@gls@doentrydef
2643      }

```

1.10.1 Links to glossary entries

The links to glossary entries all have a first optional argument that can be used to change the format and counter of the associated entry number. Except for `\glslink` and `\glsdisp`, the commands like `\gls` have a final optional argument that can be used to insert additional text in the link (this will usually be appended, but can be redefined using `\defentryfmt`). It goes against the \TeX norm to have an optional argument after the mandatory arguments, but it makes more sense to write, say, `\gls{label}[’s]` rather than, say, `\gls[append=’s]{label}`. Since these control sequences are defined to include the final square bracket, spaces will be ignored after them. This is likely to lead to confusion as most users would not expect, say, `\gls{<label>}` to ignore following spaces, so `\new@ifnextchar` from the package is required.

The following keys can be used in the first optional argument. The counter key checks that the value is the name of a valid counter.

```

2644 \define@key{glslink}{counter}{%
2645   \ifcsundef{c@#1}%
2646   {%
2647     \PackageError{glossaries}%
2648     {There is no counter called ‘#1’}%
2649     {%
2650       The counter key should have the name of a valid counter
2651       as its value%
2652     }%
2653   }%
2654   {%
2655     \def\@gls@counter{#1}%
2656   }%
2657 }

```

The value of the format key should be the name of a command (without the initial backslash) that has a single mandatory argument which can be used to

format the associated entry number.

```
2658 \define@key{glslink}{format}{%
2659   \def\@glsnumberformat{#1}}
```

The hyper key is a boolean key, it can either have the value true or false, and indicates whether or not to make a hyperlink to the relevant glossary entry. If hyper is false, an entry will still be made in the glossary, but the given text won't be a hyperlink.

```
2660 \define@boolkey{glslink}{hyper}[true]{}
```

Initialise hyper key.

```
2661 \ifdef{\hyperlink}{\KV@glslink@hypertrue}{\KV@glslink@hyperfalse}
```

The local key is a boolean key. If true this indicates that commands such as \gls should only do a local reset rather than a global one.

```
2662 \define@boolkey{glslink}{local}[true]{}
```

The original \glsifhyper command isn't particularly useful as it makes more sense to check the actual hyperlink setting rather than testing whether the starred or unstarred version has been used. Therefore, as from version 4.08, \glsifhyper is deprecated in favour of \glsifhyperon. In case there is a particular need to know whether the starred or unstarred version was used, provide a new command that determines whether the *-version, +-version or unmodified version was used.

`\glslinkvar{<unmodified case>}{<star case>}{<plus case>}`

\glslinkvar Initialise to unmodified case.

```
2663 \newcommand*{\glslinkvar}[3]{#1}
```

\glsifhyper Now deprecated.

```
2664 \newcommand*{\glsifhyper}[2]{%
2665   \glslinkvar{#1}{#2}{#1}%
2666   \GlossariesWarning{\string\glsifhyper\space is deprecated. Did
2667     you mean \string\glsifhyperon\space or \string\glslinkvar?}%
2668 }
```

\@gls@hyp@opt Used by the commands such as \glslink to determine whether to modify the hyper option.

```
2669 \newcommand*{\@gls@hyp@opt}[1]{%
2670   \let\glslinkvar\@firstofthree
2671   \let\@gls@hyp@opt@cs#1\relax
2672   \ifstar{\s@gls@hyp@opt}%
2673   {\@ifnextchar+{\@firstoftwo{\p@gls@hyp@opt}}{#1}}%
2674 }
```

\s@gls@hyp@opt Starred version

```
2675 \newcommand*{\s@gls@hyp@opt}[1][ ]{%
```

```

2676 \let\glslinkvar\@secondofthree
2677 \@gls@hyp@opt@cs[hyper=false,#1]}

```

\p@gls@hyp@opt Plus version

```

2678 \newcommand*{\p@gls@hyp@opt}[1] [] {%
2679 \let\glslinkvar\@thirdofthree
2680 \@gls@hyp@opt@cs[hyper=true,#1]}

```

Syntax:

```
\glslink[<options>]{<label>}{<text>}
```

Display *<text>* in the document, and add the entry information for *<label>* into the relevant glossary. The optional argument should be a key value list using the `\glslink` keys defined above.

There is also a starred version:

```
\glslink*[<options>]{<label>}{<text>}
```

which is equivalent to `\glslink[hyper=false,<options>]{<label>}{<text>}`

First determine which version is being used:

\glslink

```

2681 \newrobustcmd*{\glslink}{%
2682 \@gls@hyp@opt\@gls@@link
2683 }

```

\@gls@@link The main part of the business is in `\@gls@link` which shouldn't check if the term is defined as it's called by `\gls` etc which also perform that check.

```

2684 \newcommand*{\@gls@@link}[3] [] {%
2685 \ifglsentryexists{#2}%
2686 {%
2687 \let\do@gls@link@checkfirsthyper\relax
2688 \@gls@link[#1]{#2}{#3}%
2689 }{%
2690 \PackageError{glossaries}{Glossary entry ‘#2’ has not been
2691 defined}{You need to define a glossary entry before you
2692 can use it.}%

```

Display the specified text. (The entry doesn't exist so there's nothing to link it to.)

```

2693 \glstextformat{#3}%
2694 }%
2695 }

```

\link@checkfirsthyper Check for first use and switch off hyper key if hyperlink not wanted. (Should be off if first use and hyper=false is on or if first use and both the entry is in an

acronym list and the acrfootnote setting is on.) This assumes the glossary type is stored in `\glstype` and the label is stored in `\glslabel`.

```

2696 \newcommand*{\@gls@link@checkfirsthyper}{%
2697   \ifglsused{\glslabel}%
2698   {%
2699   }%
2700   {%
2701     \gls@checkisacronymlist\glstype
2702     \ifglshyperfirst
2703     \if@glsisacronymlist
2704     \ifglsacrfootnote
2705       \KV@glslink@hyperfalse
2706     \fi
2707   \fi
2708   \else
2709     \KV@glslink@hyperfalse
2710   \fi
2711 }%

```

Allow user to hook into this

```

2712 \glslinkcheckfirsthyperhook
2713 }

```

`checkfirsthyperhook` Allow used to hook into the `\gls@link@checkfirsthyper` macro

```

2714 \newcommand*{\glslinkcheckfirsthyperhook}{}

```

`\@gls@link`

```

2715 \def\@gls@link[#1]#2#3{%

```

Inserting `\leavevmode` suggested by Donald Arseneau (avoids problem with `tabularx`).

```

2716   \leavevmode
2717   \edef\glslabel{\glsdetoklabel{#2}}%

```

Save options in `\@gls@link@opts` and label in `\@gls@link@label`

```

2718   \def\@gls@link@opts{#1}%
2719   \let\@gls@link@label\glslabel

```

```

2720   \def\@glsnumberformat{\glsnumberformat}%
2721   \edef\@gls@counter{\csname glo@\glslabel @counter\endcsname}%

```

If this is in one of the “nohypertypes” glossaries, suppress the hyperlink by default

```

2722   \edef\glstype{\csname glo@\glslabel @type\endcsname}%

```

Save original setting

```

2723   \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper

```

Switch off hyper setting if the glossary type has been identified in `nohyperlist`.

```

2724   \expandafter\DTLifinlist\expandafter
2725   {\glstype}{\@gls@nohyperlist}%

```

```

2726    {%
2727        \KV@glslink@hyperfalse
2728    }%
2729    {%
2730    }%

```

Macros must set this before calling `\@gls@link`. The commands that check the first use flag should set this to `\@gls@link@checkfirsthyper` otherwise it should be set to `\relax`.

```

2731    \do@glslink@checkfirsthyper
2732    \setkeys{glslink}{#1}%

Define \glsifhyperon
2733    \ifKV@glslink@hyper
2734        \let\glsifhyperon\@firstoftwo
2735    \else
2736        \let\glsifhyperon\@secondoftwo
2737    \fi

```

Store the entry's counter in `\theglentrycounter`

```

2738    \@gls@saveentrycounter

```

Define sort key if necessary:

```

2739    \@gls@setsort{\glslabel}%

(De-tok'ing done by \@do@wrglossary)
2740    \@do@wrglossary{#2}%
2741    \ifKV@glslink@hyper
2742        \@glslink{\glolinkprefix\glslabel}{\glstextformat{#3}}%
2743    \else

2744        \glstextformat{#3}%
2745    \fi

```

Restore original setting

```

2746    \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper
2747 }

```

`\glolinkprefix`

```

2748 \newcommand*{\glolinkprefix}{glo:}

```

`\glentrycounter` Set default value of entry counter

```

2749 \def\glentrycounter{\glscounter}%

```

`\@saveentrycounter` Need to check if using equation counter in align environment:

```

2750 \newcommand*{\@gls@saveentrycounter}{%
2751     \def\@gls@Hcounter{}%

Are we using equation counter?
2752     \ifthenelse{\equal{\@gls@counter}{equation}}{%
2753     {

```

If we're in align environment, \xatlevel@ will be defined. (Can't test for \@currentenv as may be inside an inner environment.)

```

2754 \ifcsundef{xatlevel@}%
2755 {%
2756 \edef\theglsentrycounter{\expandafter\noexpand
2757 \csname the\@gls@counter\endcsname}%
2758 }%
2759 {%
2760 \ifx\xatlevel@\@empty
2761 \edef\theglsentrycounter{\expandafter\noexpand
2762 \csname the\@gls@counter\endcsname}%
2763 \else
2764 \savecounters@
2765 \advance\c@equation by 1\relax
2766 \edef\theglsentrycounter{\csname the\@gls@counter\endcsname}%

```

Check if hyperref version of this counter

```

2767 \ifcsundef{theH\@gls@counter}%
2768 {%
2769 \def\@gls@Hcounter{\theglsentrycounter}%
2770 }%
2771 {%
2772 \def\@gls@Hcounter{\csname theH\@gls@counter\endcsname}%
2773 }%
2774 \protected@edef\theHglentrycounter{\@gls@Hcounter}%
2775 \restorecounters@
2776 \fi
2777 }%
2778 }%
2779 {%

```

Not using equation counter so no special measures:

```

2780 \edef\theglsentrycounter{\expandafter\noexpand
2781 \csname the\@gls@counter\endcsname}%
2782 }%

```

Check if hyperref version of this counter

```

2783 \ifx\@gls@Hcounter\@empty
2784 \ifcsundef{theH\@gls@counter}%
2785 {%
2786 \def\theHglentrycounter{\theglsentrycounter}%
2787 }%
2788 {%
2789 \protected@edef\theHglentrycounter{\expandafter\noexpand
2790 \csname theH\@gls@counter\endcsname}%
2791 }%
2792 \fi
2793 }

```


`\@set@glo@numformat` Set the formatting information in the format required by `makeindex`. The first argument is the format specified by the user (via the format key), the second argument is the name of the counter used to indicate the location, the third argument is a control sequence which stores the required format and the fourth argument (new to v3.0) is the hyper-prefix.

```

2794 \def\@set@glo@numformat#1#2#3#4{%
2795   \expandafter\@glo@check@mkidxrangechar#3\@nil
2796   \protected@edef#1{%
2797     \@glo@prefix setentrycounter[#4]{#2}%
2798     \expandafter\string\csname\@glo@suffix\endcsname
2799   }%
2800   \@gls@checkmkidxchars#1%
2801 }

```

Check to see if the given string starts with a (or). If it does set `\@glo@prefix` to the starting character, and `\@glo@suffix` to the rest (or `glsnumberformat` if there is nothing else), otherwise set `\@glo@prefix` to nothing and `\@glo@suffix` to all of it.

```

2802 \def\@glo@check@mkidxrangechar#1#2\@nil{%
2803   \if#1(\relax
2804     \def\@glo@prefix{(%
2805     \if\relax#2\relax
2806       \def\@glo@suffix{glsnumberformat}%
2807     \else
2808       \def\@glo@suffix{#2}%
2809     \fi
2810   \else
2811     \if#1)\relax
2812     \def\@glo@prefix{)%
2813     \if\relax#2\relax
2814       \def\@glo@suffix{glsnumberformat}%
2815     \else
2816       \def\@glo@suffix{#2}%
2817     \fi
2818   \else
2819     \def\@glo@prefix{}\def\@glo@suffix{#1#2}%
2820   \fi
2821 \fi}

```

`\@gls@escbsdq` Escape backslashes and double quote marks. The argument must be a control sequence.

```

2822 \newcommand*\@gls@escbsdq[1]{%
2823   \def\@gls@checkedmkidx{%
2824     \let\gls@xdystring=#1\relax
2825     \@onelevel@sanitize\gls@xdystring
2826     \edef\do@gls@xdycheckbackslash{%
2827       \noexpand\@gls@xdycheckbackslash\gls@xdystring\noexpand\@nil
2828       \@backslashchar\@backslashchar\noexpand\null}%

```

```

2829 \do@gl@xdycheckbackslash
2830 \expandafter\@gl@updatechecked\@gl@checkedmkidx{\gl@xdystring}%
2831 \def\@gl@checkedmkidx{%
2832 \expandafter\@gl@xdycheckquote\gl@xdystring\@nil""\null
2833 \expandafter\@gl@updatechecked\@gl@checkedmkidx{\gl@xdystring}%

  Unsanitize \gl@numberpage, \gl@alphpage, \gl@Alphpage and \gl@romanpage
  (thanks to David Carlisle for the suggestion.)

2834 \@for\@gl@tmp:=\gl@protected@pagefmts\do
2835 {%
2836 \edef\@gl@sanitized@tmp{\expandafter\@gobble\string\\expandonce\@gl@tmp}%
2837 \@onelevel@sanitize\@gl@sanitized@tmp
2838 \edef\gl@dostubst{%
2839 \noexpand\DTLsubstituteall\noexpand\gl@xdystring
2840 {\@gl@sanitized@tmp}{\expandonce\@gl@tmp}%
2841 }%
2842 \gl@dostubst
2843 }%

  Assign to required control sequence
2844 \let#1=\gl@xdystring
2845 }

```

Catch special characters (argument must be a control sequence):

gl@checkmkidxchars

```

2846 \newcommand{\@gl@checkmkidxchars}[1]{%
2847 \ifgl@xindy
2848 \@gl@escbsdq{#1}%
2849 \else
2850 \def\@gl@checkedmkidx{%
2851 \expandafter\@gl@checkquote#1\@nil""\null
2852 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
2853 \def\@gl@checkedmkidx{%
2854 \expandafter\@gl@checkescquote#1\@nil\""\null
2855 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
2856 \def\@gl@checkedmkidx{%
2857 \expandafter\@gl@checkescactual#1\@nil\?\?\null
2858 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
2859 \def\@gl@checkedmkidx{%
2860 \expandafter\@gl@checkactual#1\@nil??\null
2861 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
2862 \def\@gl@checkedmkidx{%
2863 \expandafter\@gl@checkbar#1\@nil||\null
2864 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
2865 \def\@gl@checkedmkidx{%
2866 \expandafter\@gl@checkescbar#1\@nil\\|\null
2867 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
2868 \def\@gl@checkedmkidx{%
2869 \expandafter\@gl@checklevel#1\@nil!!\null

```

```

2870 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2871 \fi
2872 }

```

Update the control sequence and strip trailing \@nil:

\@gls@updatechecked

```

2873 \def\@gls@updatechecked#1\@nil#2{\def#2{#1}}

```

\@gls@tmpb Define temporary token

```

2874 \newtoks\@gls@tmpb

```

\@gls@checkquote Replace " with "" since " is a makeindex special character.

```

2875 \def\@gls@checkquote#1"#2"#3\null{%
2876 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2877 \toks@={#1}%
2878 \ifx\null#2\null
2879 \ifx\null#3\null
2880 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2881 \def\@gls@checkquote{\relax}%
2882 \else
2883 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2884 \@gls@quotechar\@gls@quotechar\@gls@quotechar\@gls@quotechar}%
2885 \def\@gls@checkquote{\@gls@checkquote#3\null}%
2886 \fi
2887 \else
2888 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2889 \@gls@quotechar\@gls@quotechar}%
2890 \ifx\null#3\null
2891 \def\@gls@checkquote{\@gls@checkquote#2""\null}%
2892 \else
2893 \def\@gls@checkquote{\@gls@checkquote#2"#3\null}%
2894 \fi
2895 \fi
2896 \@gls@checkquote
2897 }

```

\@gls@checkescquote Do the same for \":

```

2898 \def\@gls@checkescquote#1\"#2\"#3\null{%
2899 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2900 \toks@={#1}%
2901 \ifx\null#2\null
2902 \ifx\null#3\null
2903 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2904 \def\@gls@checkescquote{\relax}%
2905 \else
2906 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2907 \@gls@quotechar\string\" \@gls@quotechar
2908 \@gls@quotechar\string\" \@gls@quotechar}%

```

```

2909 \def\@gls@checkescquote{\@gls@checkescquote#3\null}%
2910 \fi
2911 \else
2912 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2913 \@gls@quotearchar\string"\@gls@quotearchar}%
2914 \ifx\null#3\null
2915 \def\@gls@checkescquote{\@gls@checkescquote#2\""\null}%
2916 \else
2917 \def\@gls@checkescquote{\@gls@checkescquote#2\">#3\null}%
2918 \fi
2919 \fi
2920 \@gls@checkescquote
2921 }

```

`\@gls@checkescactual` Similarly for `\?` (which is replaces `@` as `makeindex`'s special character):

```

2922 \def\@gls@checkescactual#1\?#2\?#3\null{%
2923 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2924 \toks@={#1}%
2925 \ifx\null#2\null
2926 \ifx\null#3\null
2927 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2928 \def\@gls@checkescactual{\relax}%
2929 \else
2930 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2931 \@gls@quotearchar\string"\@gls@actualchar
2932 \@gls@quotearchar\string"\@gls@actualchar}%
2933 \def\@gls@checkescactual{\@gls@checkescactual#3\null}%
2934 \fi
2935 \else
2936 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2937 \@gls@quotearchar\string"\@gls@actualchar}%
2938 \ifx\null#3\null
2939 \def\@gls@checkescactual{\@gls@checkescactual#2\?\?\null}%
2940 \else
2941 \def\@gls@checkescactual{\@gls@checkescactual#2\?#3\null}%
2942 \fi
2943 \fi
2944 \@gls@checkescactual
2945 }

```

`\@gls@checkescbar` Similarly for `\|`:

```

2946 \def\@gls@checkescbar#1\|#2\|#3\null{%
2947 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2948 \toks@={#1}%
2949 \ifx\null#2\null
2950 \ifx\null#3\null
2951 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2952 \def\@gls@checkescbar{\relax}%
2953 \else

```

```

2954 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2955 \@gls@quotearchar\string\"@gls@encapchar
2956 \@gls@quotearchar\string\"@gls@encapchar}%
2957 \def\@@gls@checkescbar{\@gls@checkescbar#3\null}%
2958 \fi
2959 \else
2960 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2961 \@gls@quotearchar\string\"@gls@encapchar}%
2962 \ifx\null#3\null
2963 \def\@@gls@checkescbar{\@gls@checkescbar#2\|\|\null}%
2964 \else
2965 \def\@@gls@checkescbar{\@gls@checkescbar#2\|#3\null}%
2966 \fi
2967 \fi
2968 \@@gls@checkescbar
2969 }

```

\@gls@checkesclevel Similarly for \!:

```

2970 \def\@gls@checkesclevel#1\!#2\!#3\null{%
2971 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2972 \toks@={#1}%
2973 \ifx\null#2\null
2974 \ifx\null#3\null
2975 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2976 \def\@@gls@checkesclevel{\relax}%
2977 \else
2978 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2979 \@gls@quotearchar\string\"@gls@levelchar
2980 \@gls@quotearchar\string\"@gls@levelchar}%
2981 \def\@@gls@checkesclevel{\@gls@checkesclevel#3\null}%
2982 \fi
2983 \else
2984 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2985 \@gls@quotearchar\string\"@gls@levelchar}%
2986 \ifx\null#3\null
2987 \def\@@gls@checkesclevel{\@gls@checkesclevel#2\!\!\null}%
2988 \else
2989 \def\@@gls@checkesclevel{\@gls@checkesclevel#2\!#3\null}%
2990 \fi
2991 \fi
2992 \@@gls@checkesclevel
2993 }

```

\@gls@checkbar and for |:

```

2994 \def\@gls@checkbar#1|#2|#3\null{%
2995 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2996 \toks@={#1}%
2997 \ifx\null#2\null
2998 \ifx\null#3\null

```

```

2999 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
3000 \def\@@gls@checkbar{\relax}%
3001 \else
3002 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3003 \@gls@quotechar\@gls@encapchar\@gls@quotechar\@gls@encapchar}%
3004 \def\@@gls@checkbar{\@gls@checkbar#3\null}%
3005 \fi
3006 \else
3007 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3008 \@gls@quotechar\@gls@encapchar}%
3009 \ifx\null#3\null
3010 \def\@@gls@checkbar{\@gls@checkbar#2||\null}%
3011 \else
3012 \def\@@gls@checkbar{\@gls@checkbar#2|#3\null}%
3013 \fi
3014 \fi
3015 \@@gls@checkbar
3016 }

```

\@gls@checklevel and for !:

```

3017 \def\@gls@checklevel#1!#2!#3\null{%
3018 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
3019 \toks@={#1}%
3020 \ifx\null#2\null
3021 \ifx\null#3\null
3022 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
3023 \def\@@gls@checklevel{\relax}%
3024 \else
3025 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3026 \@gls@quotechar\@gls@levelchar\@gls@quotechar\@gls@levelchar}%
3027 \def\@@gls@checklevel{\@gls@checklevel#3\null}%
3028 \fi
3029 \else
3030 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3031 \@gls@quotechar\@gls@levelchar}%
3032 \ifx\null#3\null
3033 \def\@@gls@checklevel{\@gls@checklevel#2!!\null}%
3034 \else
3035 \def\@@gls@checklevel{\@gls@checklevel#2!#3\null}%
3036 \fi
3037 \fi
3038 \@@gls@checklevel
3039 }

```

\@gls@checkactual and for ?:

```

3040 \def\@gls@checkactual#1?#2?#3\null{%
3041 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
3042 \toks@={#1}%
3043 \ifx\null#2\null

```

```

3044 \ifx\null#3\null
3045 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
3046 \def\@@gls@checkactual{\relax}%
3047 \else
3048 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3049 \@gls@quotechar\@gls@actualchar\@gls@quotechar\@gls@actualchar}%
3050 \def\@@gls@checkactual{\@gls@checkactual#3\null}%
3051 \fi
3052 \else
3053 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3054 \@gls@quotechar\@gls@actualchar}%
3055 \ifx\null#3\null
3056 \def\@@gls@checkactual{\@gls@checkactual#2??\null}%
3057 \else
3058 \def\@@gls@checkactual{\@gls@checkactual#2?#3\null}%
3059 \fi
3060 \fi
3061 \@@gls@checkactual
3062 }

```

\@gls@xdycheckquote As before but for use with xindy

```

3063 \def\@gls@xdycheckquote#1"#2"#3\null{%
3064 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
3065 \toks@={#1}%
3066 \ifx\null#2\null
3067 \ifx\null#3\null
3068 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
3069 \def\@@gls@xdycheckquote{\relax}%
3070 \else
3071 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3072 \string\}\string\}%
3073 \def\@@gls@xdycheckquote{\@gls@xdycheckquote#3\null}%
3074 \fi
3075 \else
3076 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3077 \string\}%
3078 \ifx\null#3\null
3079 \def\@@gls@xdycheckquote{\@gls@xdycheckquote#2""\null}%
3080 \else
3081 \def\@@gls@xdycheckquote{\@gls@xdycheckquote#2"#3\null}%
3082 \fi
3083 \fi
3084 \@@gls@xdycheckquote
3085 }

```

s@xdycheckbackslash Need to escape all backslashes for xindy. Define command that will define

```

\@gls@xdycheckbackslash
3086 \edef\def\@gls@xdycheckbackslash{%
3087 \noexpand\def\noexpand\@gls@xdycheckbackslash##1\@backslashchar

```

```

3088    ##2\@backslashchar##3\@noexpand\null{%
3089    \noexpand\@gls@tmpb=\noexpand\expandafter
3090    {\noexpand\@gls@checkedmkidx}%
3091    \noexpand\toks@={##1}%
3092    \noexpand\ifx\noexpand\null##2\noexpand\null
3093    \noexpand\ifx\noexpand\null##3\noexpand\null
3094    \noexpand\edef\noexpand\@gls@checkedmkidx{%
3095        \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@}%
3096    \noexpand\def\noexpand\@gls@xdycheckbackslash{\relax}%
3097    \noexpand\else
3098    \noexpand\edef\noexpand\@gls@checkedmkidx{%
3099        \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
3100        \@backslashchar\@backslashchar\@backslashchar\@backslashchar}%
3101    \noexpand\def\noexpand\@gls@xdycheckbackslash{%
3102        \noexpand\@gls@xdycheckbackslash##3\noexpand\null}%
3103    \noexpand\fi
3104    \noexpand\else
3105    \noexpand\edef\noexpand\@gls@checkedmkidx{%
3106        \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
3107        \@backslashchar\@backslashchar}%
3108    \noexpand\ifx\noexpand\null##3\noexpand\null
3109    \noexpand\def\noexpand\@gls@xdycheckbackslash{%
3110        \noexpand\@gls@xdycheckbackslash##2\@backslashchar
3111        \@backslashchar\noexpand\null}%
3112    \noexpand\else
3113    \noexpand\def\noexpand\@gls@xdycheckbackslash{%
3114        \noexpand\@gls@xdycheckbackslash##2\@backslashchar
3115        ##3\noexpand\null}%
3116    \noexpand\fi
3117    \noexpand\fi
3118    \noexpand\@gls@xdycheckbackslash
3119    }%
3120 }

```

Now go ahead and define \@gls@xdycheckbackslash

```

3121 \def@gls@xdycheckbackslash

```

\glsdohypertarget

```

3122 \newlength\gls@tmplen
3123 \newcommand*\glsdohypertarget[2]{%
3124     \settoheight{\gls@tmplen}{#2}%
3125     \raisebox{\gls@tmplen}{\hypertarget{#1}}{#2}%
3126 }

```

\glsdohyperlink

```

3127 \newcommand*\glsdohyperlink[2]{\hyperlink{#1}{#2}}

```

\@glslink If \hyperlink is not defined \@glslink ignores its first argument and just does the second argument, otherwise it is equivalent to \hyperlink.


```

3128 \ifcsundef{hyperlink}%
3129 {%
3130   \let\@glslink\@secondoftwo
3131 }%
3132 {%
3133   \let\@glslink\glsdohyperlink
3134 }

```

`\@gls target` If `\hypertarget` is not defined, `\@gls target` ignores its first argument and just does the second argument, otherwise it is equivalent to `\hypertarget`.

```

3135 \ifcsundef{hypertarget}%
3136 {%
3137   \let\@gls target\@secondoftwo
3138 }%
3139 {%
3140   \let\@gls target\glsdohypertarget
3141 }

```

Glossary hyperlinks can be disabled using `\glsdisablehyper` (effect can be localised):

`\glsdisablehyper`

```

3142 \newcommand{\glsdisablehyper}{%
3143   \KV@glslink@hyperfalse
3144   \let\@glslink\@secondoftwo
3145   \let\@gls target\@secondoftwo
3146 }

```

Glossary hyperlinks can be enabled using `\glsenablehyper` (effect can be localised):

`\glsenablehyper`

```

3147 \newcommand{\glsenablehyper}{%
3148   \KV@glslink@hypertrue
3149   \let\@glslink\glsdohyperlink
3150   \let\@gls target\glsdohypertarget
3151 }

```

Provide some convenience commands if not already defined:

```

3152 \providecommand{\@firstofthree}[3]{#1}
3153 \providecommand{\@secondofthree}[3]{#2}

```

Syntax:

<code>\gls[<i><options></i>]{<i><label></i>}[<i><insert text></i>]</code>

Link to glossary entry using singular form. The link text is taken from the value of the text or first keys used when the entry was defined.

The first optional argument is a key-value list, the same as `\glslink`, the mandatory argument is the entry label. After the mandatory argument, there is another optional argument to insert extra text in the link text (the location of the inserted text is governed by `\glsdisplay` and `\glsdisplayfirst`). As with `\glslink` there is a starred version which is the same as the unstarred version but with the `hyper` key set to false. (Additional options can also be specified in the first optional argument.)

First determine which version is being used:

```
\gls
3154 \newrobustcmd*{\gls}{\@gls@hyp@opt\@gls}
```

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

```
\@gls
3155 \newcommand*{\@gls}[2] [] {%
3156   \new@ifnextchar {\@gls@{#1}{#2}}{\@gls@{#1}{#2} []}%
3157 }
```

`\@gls@` Read in the final optional argument:

```
3158 \def\@gls@#1#2[#3]{%
3159   \glsdoifexists{#2}%
3160   {%
3161     \let\do@gls@link@checkfirsthyper\@gls@link@checkfirsthyper

3162     \let\glsifplural\@secondoftwo
3163     \let\glsupcase\@firstofthree
3164     \let\glscustomtext\@empty
3165     \def\glsinsert{#3}%
```

Determine what the link text should be (this is stored in `\@glo@text`) Note that `\@gls@link` sets `\glstyp`.

```
3166   \def\@glo@text{\csname gls@\glstyp @entryfmt\endcsname}%
```

Call `\@gls@link`. If footnote package option has been used and the glossary type is `\acronymtype`, suppress hyperlink for first use. Likewise if the `hyperfirst=false` package option is used.

```
3167   \@gls@link[#1]{#2}{\@glo@text}%
```

Indicate that this entry has now been used

```
3168   \ifKV@glslink@local
3169     \glslocalunset{#2}%
3170   \else
3171     \glsunset{#2}%
3172   \fi
3173 }%
3174 }
```

`\Gls` behaves like `\gls`, but the first letter of the link text is converted to uppercase (note that if the first letter has an accent, the accented letter will need to be grouped when you define the entry). It is mainly intended for terms that start a sentence:

`\Gls`

```
3175 \newrobustcmd*{\Gls}{\@gls@hyp@opt\@Gls}
```

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

```
3176 \newcommand*{\@Gls}[2] [] {%
3177   \new@ifnextchar[{\@Gls@{#1}{#2}}{\@Gls@{#1}{#2} []}]%
3178 }
```

`\@Gls@` Read in the final optional argument:

```
3179 \def\@Gls@#1#2[#3]{%
3180   \glsdoifexists{#2}%
3181   {%
3182     \let\do@gls@link@checkfirsthyper\@gls@link@checkfirsthyper

3183     \let\glsifplural\@secondoftwo
3184     \let\glscapscase\@secondofthree
3185     \let\glscustomtext\@empty
3186     \def\glsinsert{#3}%
```

Determine what the link text should be (this is stored in `\@glo@text`) Note that `\@gls@link` sets `\glstype`.

```
3187   \def\@glo@text{\csname gls@\glstype @entryfmt\endcsname}%
```

Call `\@gls@link` If footnote package option has been used and the glossary type is `\acronymtype`, suppress hyperlink for first use. Likewise if the `hyperfirst=false` package option is used.

```
3188   \@gls@link[#1]{#2}{\@glo@text}%
```

Indicate that this entry has now been used

```
3189   \ifKV@gls@link@local
3190     \glslocalunset{#2}%
3191   \else
3192     \glsunset{#2}%
3193   \fi
3194 }%
3195 }
```

`\GLS` behaves like `\gls`, but the link text is converted to uppercase:

`\GLS`

```
3196 \newrobustcmd*{\GLS}{\@gls@hyp@opt\@GLS}
```

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

```
3197 \newcommand*{\@GLS}[2] [] {%
3198   \new@ifnextchar[{\@GLS@{#1}{#2}}{\@GLS@{#1}{#2} []}%
3199 }
```

`\@GLS@` Read in the final optional argument:

```
3200 \def\@GLS@#1#2[#3] {%
3201   \glsdoifexists{#2}%
3202   {%
3203     \let\do@glsl@link@checkfirsthyper\@glsl@link@checkfirsthyper
3204     \let\glsl@link@secondoftwo
3205     \let\glsl@link@thirdofthree
3206     \let\glsl@link@empty
3207     \def\glsl@link{#3}%
3208     \def\@glo@text{\csname gls@\glstype @entryfmt\endcsname}%
3209     Call \@glsl@link If footnote package option has been used and the glossary
3210     type is \acronymtype, suppress hyperlink for first use. Likewise if the hyper-
3211     first=false package option is used.
3212     \@glsl@link[#1]{#2}{\@glo@text}%
3213     Indicate that this entry has now been used
3214     \ifKV@glsl@link@local
3215       \glsl@link@localunset{#2}%
3216     \else
3217       \glsl@link@unset{#2}%
3218     \fi
3219   }%
3220 }
```

Determine what the link text should be (this is stored in `\@glo@text`). Note that `\@glsl@link` sets `\glstype`.

```
3208 \def\@glo@text{\csname gls@\glstype @entryfmt\endcsname}%
```

Call `\@glsl@link` If footnote package option has been used and the glossary type is `\acronymtype`, suppress hyperlink for first use. Likewise if the `hyperfirst=false` package option is used.

```
3209 \@glsl@link[#1]{#2}{\@glo@text}%
```

Indicate that this entry has now been used

```
3210 \ifKV@glsl@link@local
3211   \glsl@link@localunset{#2}%
3212 \else
3213   \glsl@link@unset{#2}%
3214 \fi
3215 }%
3216 }
```

`\glspl` behaves in the same way as `\gls` except it uses the plural form.

`\glspl`

```
3217 \newrobustcmd*{\glspl}{\@glsl@hyp@opt\@glspl}
```

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

```
3218 \newcommand*{\@glspl}[2] [] {%
3219   \new@ifnextchar[{\@glspl@{#1}{#2}}{\@glspl@{#1}{#2} []}%
3220 }
```

`\@glspl@` Read in the final optional argument:

```
3221 \def\@glspl@#1#2[#3] {%
3222   \glsdoifexists{#2}%
3223   {%
3224     \let\do@glsl@link@checkfirsthyper\@glsl@link@checkfirsthyper
```

```

3225 \let\glsifplural\@firstoftwo
3226 \let\glscapscase\@firstofthree
3227 \let\glscustomtext\@empty
3228 \def\glsinsert{#3}%

```

Determine what the link text should be (this is stored in \@glo@text) Note that \@gls@link sets \glstype.

```

3229 \def\@glo@text{\csname gls@\glstype @entryfmt\endcsname}%

```

Call \@gls@link. If footnote package option has been used and the glossary type is \acronymtype, suppress hyperlink for first use. Likewise if the hyperfirst=false package option is used.

```

3230 \@gls@link[#1]{#2}{\@glo@text}%

```

Indicate that this entry has now been used

```

3231 \ifKV@glslink@local
3232 \glslocalunset{#2}%
3233 \else
3234 \glsunset{#2}%
3235 \fi
3236 }%
3237 }

```

\Glspl behaves in the same way as \glspl, except that the first letter of the link text is converted to uppercase (as with \Gls, if the first letter has an accent, it will need to be grouped).

\Glspl

```

3238 \newrobustcmd*{\Glspl}{\@gls@hyp@opt\@Glspl}

```

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

```

3239 \newcommand*{\@Glspl}[2][]{%
3240 \new@ifnextchar[\@Glspl@{#1}{#2}]{\@Glspl@{#1}{#2}[]}%
3241 }

```

\@Glspl@ Read in the final optional argument:

```

3242 \def\@Glspl@#1#2[#3]{%
3243 \glsdoifexists{#2}%
3244 {%
3245 \let\do@gls@link@checkfirsthyper\@gls@link@checkfirsthyper

3246 \let\glsifplural\@firstoftwo
3247 \let\glscapscase\@secondofthree
3248 \let\glscustomtext\@empty
3249 \def\glsinsert{#3}%

```

Determine what the link text should be (this is stored in \@glo@text). This needs to be expanded so that the \@glo@text can be passed to \xmakefirstuc.

Note that \@gls@link sets \glstype.

```

3250 \def\@glo@text{\csname gls@\glstype @entryfmt\endcsname}%

```

Call `\@gls@link`. If footnote package option has been used and the glossary type is `\acronymtype`, suppress hyperlink for first use. Likewise if the `hyperfirst=false` package option is used.

```
3251 \@gls@link[#1]{#2}{\@glo@text}%
```

Indicate that this entry has now been used

```
3252 \ifKV@glslink@local
3253 \glslocalunset{#2}%
3254 \else
3255 \glsunset{#2}%
3256 \fi
3257 }%
3258 }
```

`\GLSp1` behaves like `\glsp1` except that all the link text is converted to uppercase.

`\GLSp1`

```
3259 \newrobustcmd*{\GLSp1}{\@gls@hyp@opt\@GLSp1}
```

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

```
3260 \newcommand*{\@GLSp1}[2][{}]{%
3261 \new@ifnextchar[{\@GLSp1@{#1}{#2}}{\@GLSp1@{#1}{#2}[]}%
3262 }
```

`\@GLSp1` Read in the final optional argument:

```
3263 \def\@GLSp1@#1#2[#3]{%
3264 \glsdoifexists{#2}%
3265 {%
3266 \let\do@gls@link@checkfirsthyper\@gls@link@checkfirsthyper

3267 \let\glsifplural\@firstoftwo
3268 \let\glscapscase\@thirdofthree
3269 \let\glscustomtext\@empty
3270 \def\glsinsert{#3}%
```

Determine what the link text should be (this is stored in `\@glo@text`) Note that `\@gls@link` sets `\glstype`.

```
3271 \def\@glo@text{\csname gls@\glstype @entryfmt\endcsname}%
```

Call `\@gls@link`. If footnote package option has been used and the glossary type is `\acronymtype`, suppress hyperlink for first use. Likewise if the `hyperfirst=false` package option is used.

```
3272 \@gls@link[#1]{#2}{\@glo@text}%
```

Indicate that this entry has now been used

```
3273 \ifKV@glslink@local
3274 \glslocalunset{#2}%
3275 \else
```

```

3276     \glsunset{#2}%
3277     \fi
3278 }%
3279 }

```

`\glsdisp` `\glsdisp[<options>]{<label>}{<text>}` This is like `\gls` except that the link text is provided. This differs from `\glslink` in that it uses `\glsdisplay` or `\glsdisplayfirst` and unsets the first use flag.

First determine if we are using the starred form:

```

3280 \newrobustcmd*{\glsdisp}{\@gls@hyp@opt\@glsdisp}

```

Defined the un-starred form.

`\@glsdisp`

```

3281 \newcommand*{\@glsdisp}[3][\%
3282   \glsdoifexists{#2}]{%

3283   \let\do@gls@link@checkfirsthyper\@gls@link@checkfirsthyper

3284   \let\glsifplural\@secondoftwo
3285   \let\glsupcase\@firstofthree
3286   \def\glscustomtext{#3}%
3287   \def\glsinsert{}}%

```

Determine what the link text should be (this is stored in `\@glo@text`) Note that `\@gls@link` sets `\glstyp`.

```

3288   \def\@glo@text{\csname gls@\glstyp @entryfmt\endcsname}%

```

Call `\@gls@link`. If footnote package option has been used and the glossary type is `\acronymtype`, suppress hyperlink for first use. Likewise if the `hyperfirst=false` package option is used.

```

3289   \@gls@link[#1]{#2}{\@glo@text}%

```

Indicate that this entry has now been used

```

3290   \ifKV@glslink@local
3291     \glslocalunset{#2}%
3292   \else
3293     \glsunset{#2}%
3294   \fi
3295 }%
3296 }

```

`\@gls@field@link`

```

3297 \newcommand{\@gls@field@link}[3]{%
3298   \glsdoifexists{#2}%
3299   {%
3300     \let\do@gls@link@checkfirsthyper\relax
3301     \@gls@link[#1]{#2}{#3}%
3302   }%
3303 }

```

`\glstext` behaves like `\gls` except it always uses the value given by the text key and it doesn't mark the entry as used.

`\glstext`

```
3304 \newrobustcmd*{\glstext}{\@gls@hyp@opt\@glstext}
```

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

```
3305 \newcommand*{\@glstext}[2][\%
```

```
3306 \new@ifnextchar[\@glstext@{#1}{#2}]{\@glstext@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3307 \def\@glstext@#1#2[#3]{\%
```

```
3308 \@gls@field@link{#1}{#2}{\glsentrytext{#2}#3}%
```

```
3309 }
```

`\GLStext` behaves like `\glstext` except the text is converted to uppercase.

`\GLStext`

```
3310 \newrobustcmd*{\GLStext}{\@gls@hyp@opt\@GLStext}
```

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

```
3311 \newcommand*{\@GLStext}[2][\%
```

```
3312 \new@ifnextchar[\@GLStext@{#1}{#2}]{\@GLStext@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3313 \def\@GLStext@#1#2[#3]{\%
```

```
3314 \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentrytext{#2}#3}}%
```

```
3315 }
```

`\Glstext` behaves like `\glstext` except that the first letter of the text is converted to uppercase.

`\Glstext`

```
3316 \newrobustcmd*{\Glstext}{\@gls@hyp@opt\@Glstext}
```

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

```
3317 \newcommand*{\@Glstext}[2][\%
```

```
3318 \new@ifnextchar[\@Glstext@{#1}{#2}]{\@Glstext@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3319 \def\@Glstext@#1#2[#3]{\%
```

```
3320 \@gls@field@link{#1}{#2}{\Glsentrytext{#2}#3}%
```

```
3321 }
```

`\glsfirst` behaves like `\gls` except it always uses the value given by the first key and it doesn't mark the entry as used.

`\glsfirst`

```
3322 \newrobustcmd*{\glsfirst}{\@gls@hyp@opt\@glsfirst}
```


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

```
3323 \newcommand*{\@glsfirst}[2] [] {%
3324   \new@ifnextchar[{\@glsfirst@{#1}{#2}}]{\@glsfirst@{#1}{#2} []}}
```

Read in the final optional argument:

```
3325 \def\@glsfirst@#1#2[#3] {%
3326   \@gls@field@link{#1}{#2}{\glsentryfirst{#2}#3}%
3327 }
```

\Glsfirst behaves like \glsfirst except it displays the first letter in up-percase.

\Glsfirst

```
3328 \newrobustcmd*{\Glsfirst}{\@gls@hyp@opt\@Glsfirst}
```

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

```
3329 \newcommand*{\@Glsfirst}[2] [] {%
3330   \new@ifnextchar[{\@Glsfirst@{#1}{#2}}]{\@Glsfirst@{#1}{#2} []}}
```

Read in the final optional argument:

```
3331 \def\@Glsfirst@#1#2[#3] {%
3332   \@gls@field@link{#1}{#2}{\Glsentryfirst{#2}#3}%
3333 }
```

\GLSfirst behaves like \Glsfirst except it displays the text in uppercase.

\GLSfirst

```
3334 \newrobustcmd*{\GLSfirst}{\@gls@hyp@opt\@GLSfirst}
```

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

```
3335 \newcommand*{\@GLSfirst}[2] [] {%
3336   \new@ifnextchar[{\@GLSfirst@{#1}{#2}}]{\@GLSfirst@{#1}{#2} []}}
```

Read in the final optional argument:

```
3337 \def\@GLSfirst@#1#2[#3] {%
3338   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryfirst{#2}#3}}%
3339 }
```

\glsplural behaves like \gls except it always uses the value given by the plural key and it doesn't mark the entry as used.

\glsplural

```
3340 \newrobustcmd*{\glsplural}{\@gls@hyp@opt\@glsplural}
```

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

```
3341 \newcommand*{\@glsplural}[2] [] {%
3342   \new@ifnextchar[{\@glsplural@{#1}{#2}}]{\@glsplural@{#1}{#2} []}}
```

Read in the final optional argument:

```
3343 \def\@glsplural@#1#2[#3]{%
3344   \@gls@field@link{#1}{#2}{\glsentryplural{#2}#3}%
3345 }
```

\Glsplural behaves like \glsplural except that the first letter is converted to uppercase.

\Glsplural

```
3346 \newrobustcmd*{\Glsplural}{\@gls@hyp@opt\@Glsplural}
```

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

```
3347 \newcommand*{\@Glsplural}[2][\%
3348   \new@ifnextchar[{\@Glsplural@{#1}{#2}}{\@Glsplural@{#1}{#2}}]
```

Read in the final optional argument:

```
3349 \def\@Glsplural@#1#2[#3]{%
3350   \@gls@field@link{#1}{#2}{\glsentryplural{#2}#3}%
3351 }
```

\Glsplural behaves like \glsplural except that the text is converted to uppercase.

\GLSplural

```
3352 \newrobustcmd*{\GLSplural}{\@gls@hyp@opt\@GLSplural}
```

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

```
3353 \newcommand*{\@GLSplural}[2][\%
3354   \new@ifnextchar[{\@GLSplural@{#1}{#2}}{\@GLSplural@{#1}{#2}}]
```

Read in the final optional argument:

```
3355 \def\@GLSplural@#1#2[#3]{%
3356   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryplural{#2}#3}}%
3357 }
```

\glsfirstplural behaves like \gls except it always uses the value given by the firstplural key and it doesn't mark the entry as used.

\glsfirstplural

```
3358 \newrobustcmd*{\glsfirstplural}{\@gls@hyp@opt\@glsfirstplural}
```

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

```
3359 \newcommand*{\@glsfirstplural}[2][\%
3360   \new@ifnextchar[{\@glsfirstplural@{#1}{#2}}{\@glsfirstplural@{#1}{#2}}]
```

Read in the final optional argument:

```
3361 \def\@glsfirstplural@#1#2[#3]{%
3362   \@gls@field@link{#1}{#2}{\glsentryfirstplural{#2}#3}%
3363 }
```

`\Glsfirstplural` behaves like `\glsfirstplural` except that the first letter is converted to uppercase.

`\Glsfirstplural`

```
3364 \newrobustcmd*{\Glsfirstplural}{\@gls@hyp@opt\@Glsfirstplural}
```

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

```
3365 \newcommand*{\@Glsfirstplural}[2] [] {%
```

```
3366   \new@ifnextchar[{\@Glsfirstplural@{#1}{#2}}{\@Glsfirstplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3367 \def\@Glsfirstplural@#1#2[#3] {%
```

```
3368   \@gls@field@link{#1}{#2}{\Glsentryfirstplural{#2}#3}%
```

```
3369 }
```

`\GLSfirstplural` behaves like `\glsfirstplural` except that the link text is converted to uppercase.

`\GLSfirstplural`

```
3370 \newrobustcmd*{\GLSfirstplural}{\@gls@hyp@opt\@GLSfirstplural}
```

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

```
3371 \newcommand*{\@GLSfirstplural}[2] [] {%
```

```
3372   \new@ifnextchar[{\@GLSfirstplural@{#1}{#2}}{\@GLSfirstplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3373 \def\@GLSfirstplural@#1#2[#3] {%
```

```
3374   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\Glsentryfirstplural{#2}#3}}%
```

```
3375 }
```

`\glsname` behaves like `\gls` except it always uses the value given by the name key and it doesn't mark the entry as used.

`\glsname`

```
3376 \newrobustcmd*{\glsname}{\@gls@hyp@opt\@glsname}
```

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

```
3377 \newcommand*{\@glsname}[2] [] {%
```

```
3378   \new@ifnextchar[{\@glsname@{#1}{#2}}{\@glsname@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3379 \def\@glsname@#1#2[#3] {%
```

```
3380   \@gls@field@link{#1}{#2}{\Glsentryname{#2}#3}%
```

```
3381 }
```

`\Glsname` behaves like `\glsname` except that the first letter is converted to uppercase.

`\Glsname`

```
3382 \newrobustcmd*{\Glsname}{\@gls@hyp@opt\@Glsname}
```

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

```
3383 \newcommand*{\@Glsname}[2] [] {%
3384   \new@ifnextchar[{\@Glsname@{#1}{#2}}{\@Glsname@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3385 \def\@Glsname@#1#2[#3] {%
3386   \@gls@field@link{#1}{#2}{\Glsentryname{#2}#3}%
3387 }
```

\GLSname behaves like \glsname except that the link text is converted to uppercase.

\GLSname

```
3388 \newrobustcmd*{\GLSname}{\@gls@hyp@opt\@GLSname}
```

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

```
3389 \newcommand*{\@GLSname}[2] [] {%
3390   \new@ifnextchar[{\@GLSname@{#1}{#2}}{\@GLSname@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3391 \def\@GLSname@#1#2[#3] {%
3392   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\Glsentryname{#2}#3}}%
3393 }
```

\glsdesc behaves like \gls except it always uses the value given by the description key and it doesn't mark the entry as used.

\glsdesc

```
3394 \newrobustcmd*{\glsdesc}{\@gls@hyp@opt\@glsdesc}
```

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

```
3395 \newcommand*{\@glsdesc}[2] [] {%
3396   \new@ifnextchar[{\@glsdesc@{#1}{#2}}{\@glsdesc@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3397 \def\@glsdesc@#1#2[#3] {%
3398   \@gls@field@link{#1}{#2}{\Glsentrydesc{#2}#3}%
3399 }
```

\Glsdesc behaves like \glsdesc except that the first letter is converted to uppercase.

\Glsdesc

```
3400 \newrobustcmd*{\Glsdesc}{\@gls@hyp@opt\@Glsdesc}
```

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

```
3401 \newcommand*{\@Glsdesc}[2] [] {%
3402   \new@ifnextchar[{\@Glsdesc@{#1}{#2}}{\@Glsdesc@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3403 \def\@Glsdesc@#1#2[#3]{%
3404   \@gls@field@link{#1}{#2}{\Glsentrydesc{#2}#3}%
3405 }
```

\Glsdesc behaves like \glsdesc except that the link text is converted to uppercase.

\Glsdesc

```
3406 \newrobustcmd*{\Glsdesc}{\@gls@hyp@opt\@Glsdesc}
```

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

```
3407 \newcommand*{\@Glsdesc}[2][ ]{%
3408   \new@ifnextchar[{\@Glsdesc@{#1}{#2}}{\@Glsdesc@{#1}{#2}[ ]}]
```

Read in the final optional argument:

```
3409 \def\@Glsdesc@#1#2[#3]{%
3410   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\Glsentrydesc{#2}#3}}%
3411 }
```

\glsdescplural behaves like \gls except it always uses the value given by the descriptionplural key and it doesn't mark the entry as used.

\glsdescplural

```
3412 \newrobustcmd*{\glsdescplural}{\@gls@hyp@opt\@glsdescplural}
```

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

```
3413 \newcommand*{\@glsdescplural}[2][ ]{%
3414   \new@ifnextchar[{\@glsdescplural@{#1}{#2}}{\@glsdescplural@{#1}{#2}[ ]}]
```

Read in the final optional argument:

```
3415 \def\@glsdescplural@#1#2[#3]{%
3416   \@gls@field@link{#1}{#2}{\Glsentrydescplural{#2}#3}%
3417 }
```

\Glsdescplural behaves like \glsdescplural except that the first letter is converted to uppercase.

\Glsdescplural

```
3418 \newrobustcmd*{\Glsdescplural}{\@gls@hyp@opt\@Glsdescplural}
```

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

```
3419 \newcommand*{\@Glsdescplural}[2][ ]{%
3420   \new@ifnextchar[{\@Glsdescplural@{#1}{#2}}{\@Glsdescplural@{#1}{#2}[ ]}]
```

Read in the final optional argument:

```
3421 \def\@Glsdescplural@#1#2[#3]{%
3422   \@gls@field@link{#1}{#2}{\Glsentrydescplural{#2}#3}%
3423 }
```

`\GLSdescplural` behaves like `\glsdescplural` except that the link text is converted to uppercase.

`\GLSdescplural`

```
3424 \newrobustcmd*{\GLSdescplural}{\@gls@hyp@opt\@GLSdescplural}
```

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

```
3425 \newcommand*{\@GLSdescplural}[2] [] {%
```

```
3426   \new@ifnextchar[{\@GLSdescplural@{#1}{#2}}{\@GLSdescplural@{#1}{#2} []}]
```

Read in the final optional argument:

```
3427 \def\@GLSdescplural@#1#2[#3] {%
```

```
3428   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentrydescplural{#2}{#3}}}%
```

```
3429 }
```

`\glssymbol` behaves like `\gls` except it always uses the value given by the symbol key and it doesn't mark the entry as used.

`\glssymbol`

```
3430 \newrobustcmd*{\glssymbol}{\@gls@hyp@opt\@glssymbol}
```

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

```
3431 \newcommand*{\@glssymbol}[2] [] {%
```

```
3432   \new@ifnextchar[{\@glssymbol@{#1}{#2}}{\@glssymbol@{#1}{#2} []}]
```

Read in the final optional argument:

```
3433 \def\@glssymbol@#1#2[#3] {%
```

```
3434   \@gls@field@link{#1}{#2}{\glsentrysymbol{#2}{#3}}%
```

```
3435 }
```

`\Glsymbol` behaves like `\glssymbol` except that the first letter is converted to uppercase.

`\Glsymbol`

```
3436 \newrobustcmd*{\Glsymbol}{\@gls@hyp@opt\@Glsymbol}
```

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

```
3437 \newcommand*{\@Glsymbol}[2] [] {%
```

```
3438   \new@ifnextchar[{\@Glsymbol@{#1}{#2}}{\@Glsymbol@{#1}{#2} []}]
```

Read in the final optional argument:

```
3439 \def\@Glsymbol@#1#2[#3] {%
```

```
3440   \@gls@field@link{#1}{#2}{\Glsentrysymbol{#2}{#3}}%
```

```
3441 }
```

`\GLSsymbol` behaves like `\glssymbol` except that the link text is converted to uppercase.

`\GLSsymbol`

```
3442 \newrobustcmd*{\GLSsymbol}{\@gls@hyp@opt\@GLSsymbol}
```

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

```
3443 \newcommand*{\@GLSsymbol}[2] [] {%
3444   \new@ifnextchar[{\@GLSsymbol@{#1}{#2}}{\@GLSsymbol@{#1}{#2} [] }}
```

Read in the final optional argument:

```
3445 \def\@GLSsymbol@#1#2[#3] {%
3446   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentrysymbol{#2}#3}}%
3447 }
```

`\glsymbolplural` behaves like `\gls` except it always uses the value given by the `symbolplural` key and it doesn't mark the entry as used.

`\glsymbolplural`

```
3448 \newrobustcmd*{\glsymbolplural}{\@gls@hyp@opt\@glsymbolplural}
```

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

```
3449 \newcommand*{\@glsymbolplural}[2] [] {%
3450   \new@ifnextchar[{\@glsymbolplural@{#1}{#2}}{\@glsymbolplural@{#1}{#2} [] }}
```

Read in the final optional argument:

```
3451 \def\@glsymbolplural@#1#2[#3] {%
3452   \@gls@field@link{#1}{#2}{\glsentrysymbolplural{#2}#3}%
3453 }
```

`\Glsymbolplural` behaves like `\glsymbolplural` except that the first letter is converted to uppercase.

`\Glsymbolplural`

```
3454 \newrobustcmd*{\Glsymbolplural}{\@gls@hyp@opt\@Glsymbolplural}
```

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

```
3455 \newcommand*{\@Glsymbolplural}[2] [] {%
3456   \new@ifnextchar[{\@Glsymbolplural@{#1}{#2}}{\@Glsymbolplural@{#1}{#2} [] }}
```

Read in the final optional argument:

```
3457 \def\@Glsymbolplural@#1#2[#3] {%
3458   \@gls@field@link{#1}{#2}{\Glsentrysymbolplural{#2}#3}%
3459 }
```

`\GLSsymbolplural` behaves like `\glsymbolplural` except that the link text is converted to uppercase.

`\GLSsymbolplural`

```
3460 \newrobustcmd*{\GLSsymbolplural}{\@gls@hyp@opt\@GLSsymbolplural}
```

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

```
3461 \newcommand*{\@GLSsymbolplural}[2] [] {%
3462   \new@ifnextchar[{\@GLSsymbolplural@{#1}{#2}}{\@GLSsymbolplural@{#1}{#2} [] }}
```

Read in the final optional argument:

```
3463 \def\@GLSsymbolplural@#1#2[#3]{%
3464   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentrysymbolplural{#2}#3}}%
3465 }
```

`\glsuseri` behaves like `\gls` except it always uses the value given by the `user1` key and it doesn't mark the entry as used.

`\glsuseri`

```
3466 \newrobustcmd*{\glsuseri}{\@gls@hyp@opt\@glsuseri}
```

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

```
3467 \newcommand*{\@glsuseri}[2] [] {%
3468   \new@ifnextchar[{\@glsuseri@{#1}{#2}}{\@glsuseri@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3469 \def\@glsuseri@#1#2[#3]{%
3470   \@gls@field@link{#1}{#2}{\glsentryuseri{#2}#3}%
3471 }
```

`\Glsuseri` behaves like `\glsuseri` except that the first letter is converted to uppercase.

`\Glsuseri`

```
3472 \newrobustcmd*{\Glsuseri}{\@gls@hyp@opt\@Glsuseri}
```

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

```
3473 \newcommand*{\@Glsuseri}[2] [] {%
3474   \new@ifnextchar[{\@Glsuseri@{#1}{#2}}{\@Glsuseri@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3475 \def\@Glsuseri@#1#2[#3]{%
3476   \@gls@field@link{#1}{#2}{\Glsentryuseri{#2}#3}%
3477 }
```

`\GLSuseri` behaves like `\glsuseri` except that the link text is converted to uppercase.

`\GLSuseri`

```
3478 \newrobustcmd*{\GLSuseri}{\@gls@hyp@opt\@GLSuseri}
```

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

```
3479 \newcommand*{\@GLSuseri}[2] [] {%
3480   \new@ifnextchar[{\@GLSuseri@{#1}{#2}}{\@GLSuseri@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3481 \def\@GLSuseri@#1#2[#3]{%
3482   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryuseri{#2}#3}}%
3483 }
```


`\glsuserii` behaves like `\gls` except it always uses the value given by the `user2` key and it doesn't mark the entry as used.

`\glsuserii`

```
3484 \newrobustcmd*{\glsuserii}{\@gls@hyp@opt\@glsuserii}
```

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

```
3485 \newcommand*{\@glsuserii}[2][]{%
```

```
3486 \new@ifnextchar[{\@glsuserii@{#1}{#2}}{\@glsuserii@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3487 \def\@glsuserii@#1#2[#3]{%
```

```
3488 \@gls@field@link{#1}{#2}{\glsentryuserii{#2}#3}%
```

```
3489 }
```

`\Glsuserii` behaves like `\glsuserii` except that the first letter is converted to uppercase.

`\Glsuserii`

```
3490 \newrobustcmd*{\Glsuserii}{\@gls@hyp@opt\@Glsuserii}
```

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

```
3491 \newcommand*{\@Glsuserii}[2][]{%
```

```
3492 \new@ifnextchar[{\@Glsuserii@{#1}{#2}}{\@Glsuserii@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3493 \def\@Glsuserii@#1#2[#3]{%
```

```
3494 \@gls@field@link{#1}{#2}{\Glsentryuserii{#2}#3}%
```

```
3495 }
```

`\GLSuserii` behaves like `\glsuserii` except that the link text is converted to uppercase.

`\GLSuserii`

```
3496 \newrobustcmd*{\GLSuserii}{\@gls@hyp@opt\@GLSuserii}
```

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

```
3497 \newcommand*{\@GLSuserii}[2][]{%
```

```
3498 \new@ifnextchar[{\@GLSuserii@{#1}{#2}}{\@GLSuserii@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3499 \def\@GLSuserii@#1#2[#3]{%
```

```
3500 \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryuserii{#2}#3}}%
```

```
3501 }
```

`\glsuseriii` behaves like `\gls` except it always uses the value given by the `user3` key and it doesn't mark the entry as used.

`\glsuseriii`

```
3502 \newrobustcmd*{\glsuseriii}{\@gls@hyp@opt\@glsuseriii}
```

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

```
3503 \newcommand*{\@glsuseriii}[2] [] {%
3504   \new@ifnextchar[{\@glsuseriii@{#1}{#2}}{\@glsuseriii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3505 \def\@glsuseriii@#1#2[#3]{%
3506   \@gls@field@link{#1}{#2}{\glstentryuseriii{#2}#3}%
3507 }
```

\Glsuseriii behaves like \glsuseriii except that the first letter is converted to uppercase.

\Glsuseriii

```
3508 \newrobustcmd*{\Glsuseriii}{\@gls@hyp@opt\@Glsuseriii}
```

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

```
3509 \newcommand*{\@Glsuseriii}[2] [] {%
3510   \new@ifnextchar[{\@Glsuseriii@{#1}{#2}}{\@Glsuseriii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3511 \def\@Glsuseriii@#1#2[#3]{%
3512   \@gls@field@link{#1}{#2}{\Glstentryuseriii{#2}#3}%
3513 }
```

\GLSuseriii behaves like \glsuseriii except that the link text is converted to uppercase.

\GLSuseriii

```
3514 \newrobustcmd*{\GLSuseriii}{\@gls@hyp@opt\@GLSuseriii}
```

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

```
3515 \newcommand*{\@GLSuseriii}[2] [] {%
3516   \new@ifnextchar[{\@GLSuseriii@{#1}{#2}}{\@GLSuseriii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3517 \def\@GLSuseriii@#1#2[#3]{%
3518   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glstentryuseriii{#2}#3}}%
3519 }
```

\glsuseriv behaves like \gls except it always uses the value given by the user4 key and it doesn't mark the entry as used.

\glsuseriv

```
3520 \newrobustcmd*{\glsuseriv}{\@gls@hyp@opt\@glsuseriv}
```

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

```
3521 \newcommand*{\@glsuseriv}[2] [] {%
3522   \new@ifnextchar[{\@glsuseriv@{#1}{#2}}{\@glsuseriv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3523 \def\@glsuseriv@#1#2[#3]{%
3524   \@gls@field@link{#1}{#2}{\glstryuseriv{#2}#3}%
3525 }
```

\Glsuseriv behaves like \glsuseriv except that the first letter is converted to uppercase.

\Glsuseriv

```
3526 \newrobustcmd*{\Glsuseriv}{\@gls@hyp@opt\@Glsuseriv}
```

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

```
3527 \newcommand*{\@Glsuseriv}[2][ ]{%
3528   \new@ifnextchar[{\@Glsuseriv@{#1}{#2}}{\@Glsuseriv@{#1}{#2}[ ]}]}
```

Read in the final optional argument:

```
3529 \def\@Glsuseriv@#1#2[#3]{%
3530   \@gls@field@link{#1}{#2}{\glstryuseriv{#2}#3}%
3531 }
```

\GLSuseriv behaves like \glsuseriv except that the link text is converted to uppercase.

\GLSuseriv

```
3532 \newrobustcmd*{\GLSuseriv}{\@gls@hyp@opt\@GLSuseriv}
```

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

```
3533 \newcommand*{\@GLSuseriv}[2][ ]{%
3534   \new@ifnextchar[{\@GLSuseriv@{#1}{#2}}{\@GLSuseriv@{#1}{#2}[ ]}]}
```

Read in the final optional argument:

```
3535 \def\@GLSuseriv@#1#2[#3]{%
3536   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glstryuseriv{#2}#3}}%
3537 }
```

\glsuserv behaves like \gls except it always uses the value given by the user5 key and it doesn't mark the entry as used.

\glsuserv

```
3538 \newrobustcmd*{\glsuserv}{\@gls@hyp@opt\@glsuserv}
```

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

```
3539 \newcommand*{\@glsuserv}[2][ ]{%
3540   \new@ifnextchar[{\@glsuserv@{#1}{#2}}{\@glsuserv@{#1}{#2}[ ]}]}
```

Read in the final optional argument:

```
3541 \def\@glsuserv@#1#2[#3]{%
3542   \@gls@field@link{#1}{#2}{\glstryuserv{#2}#3}%
3543 }
```


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

```
3563 \newcommand*{\@Glsuservi}[2] [] {%
3564   \new@ifnextchar[{\@Glsuservi@{#1}{#2}}{\@Glsuservi@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3565 \def\@Glsuservi@#1#2[#3] {%
3566   \@gls@field@link{#1}{#2}{\glentryuservi{#2}#3}%
3567 }
```

\GLSuservi behaves like \glsuservi except that the link text is converted to uppercase.

\GLSuservi

```
3568 \newrobustcmd*{\GLSuservi}{\@gls@hyp@opt\@GLSuservi}
```

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

```
3569 \newcommand*{\@GLSuservi}[2] [] {%
3570   \new@ifnextchar[{\@GLSuservi@{#1}{#2}}{\@GLSuservi@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3571 \def\@GLSuservi@#1#2[#3] {%
3572   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glentryuservi{#2}#3}}%
3573 }
```

Now deal with acronym related keys. First the short form:

\acrshort

```
3574 \newrobustcmd*{\acrshort}{\@gls@hyp@opt\@ns@acrshort}
```

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

```
3575 \newcommand*{\@ns@acrshort}[2] [] {%
3576   \new@ifnextchar[{\@acrshort{#1}{#2}}{\@acrshort{#1}{#2} []}]%
3577 }
```

Read in the final optional argument:

```
3578 \def\@acrshort#1#2[#3] {%
3579   \glsdoifexists{#2}%
3580   {%
3581     \let\do@gls@link@checkfirsthyper\relax
3582     \let\glsifplural\@secondoftwo
3583     \let\glscapscase\@firstofthree
3584     \let\glsinsert\@empty
3585     \def\glscustomtext{%
3586       \acronymfont{\glentryshort{#2}}#3%
3587     }%
```

Call \@gls@link Note that \@gls@link sets \glstype.

```
3588   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3589   }%
3590 }
```

\Acrshort

```
3591 \newrobustcmd*{\Acrshort}{\@gls@hyp@opt\@ns@Acrshort}
```

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

```
3592 \newcommand*{\ns@Acrshort}[2] [] {%
3593   \new@ifnextchar[{\@Acrshort{#1}{#2}}{\@Acrshort{#1}{#2} []}%
3594 }
```

Read in the final optional argument:

```
3595 \def\@Acrshort#1#2[#3] {%
3596   \glsdoifexists{#2}%
3597   {%
3598     \let\do@gls@link@checkfirsthyper\relax

3599     \def\glslabel{#2}%
3600     \let\glsifplural\@secondoftwo
3601     \let\glsapscase\@secondofthree
3602     \let\glsinsert\@empty
3603     \def\glscustomtext{%
3604       \acronymfont{\Glsentryshort{#2}}#3%
3605     }%
```

Call \@gls@link Note that \@gls@link sets \glstype.

```
3606   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3607   }%
3608 }
```

\ACRshort

```
3609 \newrobustcmd*{\ACRshort}{\@gls@hyp@opt\@ns@ACRshort}
```

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

```
3610 \newcommand*{\ns@ACRshort}[2] [] {%
3611   \new@ifnextchar[{\@ACRshort{#1}{#2}}{\@ACRshort{#1}{#2} []}%
3612 }
```

Read in the final optional argument:

```
3613 \def\@ACRshort#1#2[#3] {%
3614   \glsdoifexists{#2}%
3615   {%
3616     \let\do@gls@link@checkfirsthyper\relax
```

```

3617 \def\glslabel{#2}%
3618 \let\glsifplural\@secondoftwo
3619 \let\glscapscase\@thirdofthree
3620 \let\glsinsert\@empty
3621 \def\glscustomtext{%
3622     \mfirstucMakeUppercase{\acronymfont{\glsentryshort{#2}}#3}%
3623 }%

```

Call \@gls@link Note that \@gls@link sets \glstype.

```

3624 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3625 }%
3626 }

```

Short plural:

\acrshortpl

```

3627 \newrobustcmd*{\acrshortpl}{\@gls@hyp@opt\@ns@acrshortpl}

```

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

```

3628 \newcommand*{\ns@acrshortpl}[2] [] {%
3629     \new@ifnextchar[{\@acrshortpl{#1}{#2}}{\@acrshortpl{#1}{#2} []}%
3630 }

```

Read in the final optional argument:

```

3631 \def\@acrshortpl#1#2[#3] {%
3632     \glsdoifexists{#2}%
3633     {%
3634         \let\do@gls@link@checkfirsthyper\relax
3635
3636         \def\glslabel{#2}%
3637         \let\glsifplural\@firstoftwo
3638         \let\glscapscase\@firstofthree
3639         \let\glsinsert\@empty
3640         \def\glscustomtext{%
3641             \acronymfont{\glsentryshortpl{#2}}#3%
3642         }%
3643     }%
3644 }

```

Call \@gls@link Note that \@gls@link sets \glstype.

```

3642 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3643 }%
3644 }

```

\Acrshortpl

```

3645 \newrobustcmd*{\Acrshortpl}{\@gls@hyp@opt\@ns@Acrshortpl}

```

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

```

3646 \newcommand*{\ns@Acrshortpl}[2] [] {%
3647     \new@ifnextchar[{\@Acrshortpl{#1}{#2}}{\@Acrshortpl{#1}{#2} []}%
3648 }

```

Read in the final optional argument:

```
3649 \def\@Acrshortpl#1#2[#3]{%
3650   \glsdoifexists{#2}%
3651   {%
3652     \let\do@gl@link@checkfirsthyper\relax

3653     \def\glslabel{#2}%
3654     \let\glsifplural\@firstoftwo
3655     \let\glscapscase\@secondofthree
3656     \let\glsinsert\@empty
3657     \def\glscustomtext{%
3658       \acronymfont{\glentryshortpl{#2}}#3%
3659     }%
```

Call \@gl@link Note that \@gl@link sets \glstype.

```
3660   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3661   }%
3662 }
```

\ACRshortpl

```
3663 \newrobustcmd*{\ACRshortpl}{\@gl@hyp@opt\ns@ACRshortpl}
```

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

```
3664 \newcommand*{\ns@ACRshortpl}[2][ ]{%
3665   \new@ifnextchar[{\@ACRshortpl{#1}{#2}}{\@ACRshortpl{#1}{#2}[]}%
3666 }
```

Read in the final optional argument:

```
3667 \def\@ACRshortpl#1#2[#3]{%
3668   \glsdoifexists{#2}%
3669   {%
3670     \let\do@gl@link@checkfirsthyper\relax

3671     \def\glslabel{#2}%
3672     \let\glsifplural\@firstoftwo
3673     \let\glscapscase\@thirdofthree
3674     \let\glsinsert\@empty
3675     \def\glscustomtext{%
3676       \mfirstucMakeUppercase{\acronymfont{\glentryshortpl{#2}}#3}%
3677     }%
```

Call \@gl@link Note that \@gl@link sets \glstype.

```
3678   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3679   }%
3680 }
```

\acrlong

```
3681 \newrobustcmd*{\acrlong}{\@gl@hyp@opt\ns@acrlong}
```


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

```
3682 \newcommand*{\ns@acrlong}[2] [] {%
3683   \new@ifnextchar[{\@acrlong{#1}{#2}}{\@acrlong{#1}{#2} []}%
3684 }
```

Read in the final optional argument:

```
3685 \def\@acrlong#1#2[#3] {%
3686   \glstoifexists{#2}%
3687   {%
3688     \let\do@gl@link@checkfirsthyper\relax

3689   \def\glslabel{#2}%
3690   \let\gl@ifplural\@secondoftwo
3691   \let\glscapscase\@firstofthree
3692   \let\glinsert\@empty
```

Bug fix v4.02 removed \acronymfont from \glscustomtext (\acronymfont only designed for short form).

```
3693   \def\glscustomtext{%
3694     \glstrylong{#2}#3%
3695   }%
```

Call \@gl@link Note that \@gl@link sets \glstype.

```
3696   \@gl@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
3697   }%
3698 }
```

\Acrlong

```
3699 \newrobustcmd*{\Acrlong}{\@gl@hyp@opt\ns@Acrlong}
```

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

```
3700 \newcommand*{\ns@Acrlong}[2] [] {%
3701   \new@ifnextchar[{\@Acrlong{#1}{#2}}{\@Acrlong{#1}{#2} []}%
3702 }
```

Read in the final optional argument:

```
3703 \def\@Acrlong#1#2[#3] {%
3704   \glstoifexists{#2}%
3705   {%
3706     \let\do@gl@link@checkfirsthyper\relax

3707   \def\glslabel{#2}%
3708   \let\gl@ifplural\@secondoftwo
3709   \let\glscapscase\@secondofthree
3710   \let\glinsert\@empty
```

Bug fix v4.02 removed \acronymfont from \glscustomtext (\acronymfont only designed for short form).

```

3711 \def\glscustomtext{%
3712 \Glsentrylong{#2}#3%
3713 }%

```

Call \@gls@link. Note that \@gls@link sets \glstype.

```

3714 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3715 }%
3716 }

```

\ACRlong

```

3717 \newrobustcmd*{\ACRlong}{\@gls@hyp@opt\@ns@ACRlong}

```

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

```

3718 \newcommand*{\ns@ACRlong}[2][{}]{%
3719 \new@ifnextchar[{\@ACRlong{#1}{#2}}{\@ACRlong{#1}{#2}[]}%
3720 }

```

Read in the final optional argument:

```

3721 \def\@ACRlong#1#2[#3]{%
3722 \glsdoifexists{#2}%
3723 {%
3724 \let\do@gls@link@checkfirsthyper\relax

```

```

3725 \def\glslabel{#2}%
3726 \let\glsifplural\@secondoftwo
3727 \let\glscapscase\@thirdofthree
3728 \let\glsinsert\@empty

```

Bug fix v4.02 removed \acronymfont from \glscustomtext (\acronymfont only designed for short form).

```

3729 \def\glscustomtext{%
3730 \mfirstucMakeUppercase{\glsentrylong{#2}#3}%
3731 }%

```

Call \@gls@link. Note that \@gls@link sets \glstype.

```

3732 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3733 }%
3734 }

```

Short plural:

\acrlongpl

```

3735 \newrobustcmd*{\acrlongpl}{\@gls@hyp@opt\@ns@acrlongpl}

```

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

```

3736 \newcommand*{\ns@acrlongpl}[2][{}]{%
3737 \new@ifnextchar[{\@acrlongpl{#1}{#2}}{\@acrlongpl{#1}{#2}[]}%
3738 }

```

Read in the final optional argument:

```

3739 \def\@acrlongpl#1#2[#3]{%
3740   \glsdoifexists{#2}%
3741   {%
3742     \let\do@gl@link@checkfirsthyper\relax

3743   \def\glslabel{#2}%
3744   \let\glsifplural\@firstoftwo
3745   \let\glscapscase\@firstofthree
3746   \let\glsinsert\@empty

```

Bug fix v4.02 removed \acronymfont from \glscustomtext (\acronymfont only designed for short form).

```

3747   \def\glscustomtext{%
3748     \glentrylongpl{#2}#3%
3749   }%

```

Call \@gl@link. Note that \@gl@link sets \glstype.

```

3750   \@gl@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
3751   }%
3752 }

```

\Acrlongpl

```

3753 \newrobustcmd*{\Acrlongpl}{\@gl@hyp@opt\@ns@Acrlongpl}

```

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

```

3754 \newcommand*{\ns@Acrlongpl}[2][]{%
3755   \new@ifnextchar[{\@Acrlongpl{#1}{#2}}{\@Acrlongpl{#1}{#2}[]}%
3756 }

```

Read in the final optional argument:

```

3757 \def\@Acrlongpl#1#2[#3]{%
3758   \glsdoifexists{#2}%
3759   {%
3760     \let\do@gl@link@checkfirsthyper\relax

3761   \def\glslabel{#2}%
3762   \let\glsifplural\@firstoftwo
3763   \let\glscapscase\@secondofthree
3764   \let\glsinsert\@empty

```

Bug fix v4.02 removed \acronymfont from \glscustomtext (\acronymfont only designed for short form).

```

3765   \def\glscustomtext{%
3766     \Glsentrylongpl{#2}#3%
3767   }%

```

Call \@gl@link. Note that \@gl@link sets \glstype.

```

3768   \@gl@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3769   }%
3770 }

```

\ACRlongpl

```
3771 \newrobustcmd*{\ACRlongpl}{\@gls@hyp@opt\ns@ACRlongpl}
```

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

```
3772 \newcommand*{\ns@ACRlongpl}[2] [] {%
3773   \new@ifnextchar[{\@ACRlongpl{#1}{#2}}{\@ACRlongpl{#1}{#2} []}%
3774 }
```

Read in the final optional argument:

```
3775 \def\@ACRlongpl#1#2[#3] {%
3776   \glsdoifexists{#2}%
3777   {%
3778     \let\do@gls@link@checkfirsthyper\relax
3779     \def\glslabel{#2}%
3780     \let\glsifplural\@firstoftwo
3781     \let\glscapscase\@thirdofthree
3782     \let\glsinsert\@empty
```

Bug fix v4.02 removed \acronymfont from \glscustomtext (\acronymfont only designed for short form).

```
3783   \def\glscustomtext{%
3784     \mfirstucMakeUppercase{\glsentrylongpl{#2}{#3}%
3785   }%
```

Call \@gls@link. Note that \@gls@link sets \glstype.

```
3786   \@gls@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
3787   }%
3788 }
```

1.10.2 Displaying entry details without adding information to the glossary

These commands merely display entry information without adding entries in the associated file or having hyperlinks.

\@gls@entry@field Generic version.

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

```
3789 \newcommand*{\@gls@entry@field}[2] {%
3790   \csname glo@glsdetoklabel{#1}@#2\endcsname
3791 }
```

\glsletentryfield

\glsletentryfield{<cs>}{<label>}{<field>}

```

3792 \newcommand*\glsletentryfield}[3]{%
3793   \letcs{#1}{glo@\glsdetoklabel{#2}@#3}%
3794 }

```

`\@Gls@entry@field` Generic first letter uppercase version.

<code>\@Gls@entry@field{<label>}{<field>}</code>
--

```

3795 \newcommand*\@Gls@entry@field}[2]{%
3796   \letcs\@glo@text{glo@\glsdetoklabel{#1}@#2}%
3797   \xmakefirstuc{\@glo@text}%
3798 }

```

Get the entry name (as specified by the name key when the entry was defined). The argument is the label associated with the entry. Note that unless you used `name=false` in the `sanitize` package option you may get unexpected results if the name key contains any commands.

`\glsentryname`

```

3799 \newcommand*\glsentryname}[1]{\@Gls@entry@field{#1}{name}}

```

`\Glsentryname`

```

3800 \newrobustcmd*\Glsentryname}[1]{%
3801   \@Gls@entry@field{#1}{name}%
3802 }

```

Get the entry description (as specified by the description when the entry was defined). The argument is the label associated with the entry. Note that unless you used `description=false` in the `sanitize` package option you may get unexpected results if the description key contained any commands.

`\glsentrydesc`

```

3803 \newcommand*\glsentrydesc}[1]{\@Gls@entry@field{#1}{desc}}

```

`\Glsentrydesc`

```

3804 \newrobustcmd*\Glsentrydesc}[1]{%
3805   \@Gls@entry@field{#1}{desc}%
3806 }

```

Plural form:

`\glsentrydescplural`

```

3807 \newcommand*\glsentrydescplural}[1]{%
3808   \@Gls@entry@field{#1}{descplural}%
3809 }

```

`\Glsentrydescplural`

```
3810 \newrobustcmd*{\Glsentrydescplural}[1]{%  
3811   \@Gls@entry@field{#1}{descplural}%  
3812 }
```

Get the entry text, as specified by the text key when the entry was defined.
The argument is the label associated with the entry:

`\glsentrytext`

```
3813 \newcommand*{\glsentrytext}[1]{\@Gls@entry@field{#1}{text}}
```

`\Glsentrytext`

```
3814 \newrobustcmd*{\Glsentrytext}[1]{%  
3815   \@Gls@entry@field{#1}{text}%  
3816 }
```

Get the plural form:

`\glsentryplural`

```
3817 \newcommand*{\glsentryplural}[1]{%  
3818   \@Gls@entry@field{#1}{plural}%  
3819 }
```

`\Glsentryplural`

```
3820 \newrobustcmd*{\Glsentryplural}[1]{%  
3821   \@Gls@entry@field{#1}{plural}%  
3822 }
```

Get the symbol associated with this entry. The argument is the label associated with the entry.

`\glsentrysymbol`

```
3823 \newcommand*{\glsentrysymbol}[1]{%  
3824   \@Gls@entry@field{#1}{symbol}%  
3825 }
```

`\Glsentrysymbol`

```
3826 \newrobustcmd*{\Glsentrysymbol}[1]{%  
3827   \@Gls@entry@field{#1}{symbol}%  
3828 }
```

Plural form:

`\glsentrysymbolplural`

```
3829 \newcommand*{\glsentrysymbolplural}[1]{%  
3830   \@Gls@entry@field{#1}{symbolplural}%  
3831 }
```

lentrysymbolplural

```
3832 \newrobustcmd*{\Glsentrysymbolplural}[1]{%
3833   \@Gls@entry@field{#1}{symbolplural}%
3834 }
```

Get the entry text to be used when the entry is first used in the document (as specified by the first key when the entry was defined).

\glsentryfirst

```
3835 \newcommand*{\glsentryfirst}[1]{%
3836   \@Gls@entry@field{#1}{first}%
3837 }
```

\Glsentryfirst

```
3838 \newrobustcmd*{\Glsentryfirst}[1]{%
3839   \@Gls@entry@field{#1}{first}%
3840 }
```

Get the plural form (as specified by the firstplural key when the entry was defined).

glsentryfirstplural

```
3841 \newcommand*{\glsentryfirstplural}[1]{%
3842   \@Gls@entry@field{#1}{firstpl}%
3843 }
```

Glsentryfirstplural

```
3844 \newrobustcmd*{\Glsentryfirstplural}[1]{%
3845   \@Gls@entry@field{#1}{firstpl}%
3846 }
```

Display the glossary type with which this entry is associated (as specified by the type key used when the entry was defined)

\glsentrytype

```
3847 \newcommand*{\glsentrytype}[1]{\@Gls@entry@field{#1}{type}}
```

Display the sort text used for this entry. Note that the sort key is sanitized, so unexpected results may occur if the sort key contained commands.

\glsentrysort

```
3848 \newcommand*{\glsentrysort}[1]{%
3849   \@Gls@entry@field{#1}{sort}%
3850 }
```

\glsentryuseri Get the first user key (as specified by the user1 when the entry was defined).
The argument is the label associated with the entry.

```
3851 \newcommand*{\glsentryuseri}[1]{%
3852   \@Gls@entry@field{#1}{useri}%
3853 }
```

```

\Glsentryuseri
3854 \newrobustcmd*{\Glsentryuseri}[1]{%
3855   \@Gls@entry@field{#1}{useri}%
3856 }

\glsentryuserii  Get the second user key (as specified by the user2 when the entry was defined).
                  The argument is the label associated with the entry.
3857 \newcommand*{\glsentryuserii}[1]{%
3858   \@Gls@entry@field{#1}{userii}%
3859 }

\Glsentryuserii
3860 \newrobustcmd*{\Glsentryuserii}[1]{%
3861   \@Gls@entry@field{#1}{userii}%
3862 }

\glsentryuseriii  Get the third user key (as specified by the user3 when the entry was defined).
                   The argument is the label associated with the entry.
3863 \newcommand*{\glsentryuseriii}[1]{%
3864   \@Gls@entry@field{#1}{useriii}%
3865 }

\Glsentryuseriii
3866 \newrobustcmd*{\Glsentryuseriii}[1]{%
3867   \@Gls@entry@field{#1}{useriii}%
3868 }

\glsentryuseriv  Get the fourth user key (as specified by the user4 when the entry was defined).
                  The argument is the label associated with the entry.
3869 \newcommand*{\glsentryuseriv}[1]{%
3870   \@Gls@entry@field{#1}{useriv}%
3871 }

\Glsentryuseriv
3872 \newrobustcmd*{\Glsentryuseriv}[1]{%
3873   \@Gls@entry@field{#1}{useriv}%
3874 }

\glsentryuserv   Get the fifth user key (as specified by the user5 when the entry was defined).
                  The argument is the label associated with the entry.
3875 \newcommand*{\glsentryuserv}[1]{%
3876   \@Gls@entry@field{#1}{userv}%
3877 }

\Glsentryuserv
3878 \newrobustcmd*{\Glsentryuserv}[1]{%
3879   \@Gls@entry@field{#1}{userv}%
3880 }

```


`\glsentryuservi` Get the sixth user key (as specified by the user6 when the entry was defined).
The argument is the label associated with the entry.

```
3881 \newrobustcmd*{\glsentryuservi}[1]{%
3882   \@gls@entry@field{#1}{uservi}%
3883 }
```

`\Glsentryuservi`

```
3884 \newrobustcmd*{\Glsentryuservi}[1]{%
3885   \@Gls@entry@field{#1}{uservi}%
3886 }
```

`\glsentryshort` Get the short key (as specified by the short the entry was defined). The argument is the label associated with the entry.

```
3887 \newcommand*{\glsentryshort}[1]{\@gls@entry@field{#1}{short}}
```

`\Glsentryshort`

```
3888 \newrobustcmd*{\Glsentryshort}[1]{%
3889   \@Gls@entry@field{#1}{short}%
3890 }
```

`\glsentryshortpl` Get the short plural key (as specified by the shortplural the entry was defined).
The argument is the label associated with the entry.

```
3891 \newcommand*{\glsentryshortpl}[1]{\@gls@entry@field{#1}{shortpl}}
```

`\Glsentryshortpl`

```
3892 \newrobustcmd*{\Glsentryshortpl}[1]{%
3893   \@Gls@entry@field{#1}{shortpl}%
3894 }
```

`\glsentrylong` Get the long key (as specified by the long the entry was defined). The argument is the label associated with the entry.

```
3895 \newcommand*{\glsentrylong}[1]{\@gls@entry@field{#1}{long}}
```

`\Glsentrylong`

```
3896 \newrobustcmd*{\Glsentrylong}[1]{%
3897   \@Gls@entry@field{#1}{long}%
3898 }
```

`\glsentrylongpl` Get the long plural key (as specified by the longplural the entry was defined).
The argument is the label associated with the entry.

```
3899 \newcommand*{\glsentrylongpl}[1]{\@gls@entry@field{#1}{longpl}}
```

`\Glsentrylongpl`

```
3900 \newrobustcmd*{\Glsentrylongpl}[1]{%
3901   \@Gls@entry@field{#1}{longpl}%
3902 }
```

Short cut macros to access full form:

`\glsentryfull`

```
3903 \newcommand*\glsentryfull}[1]{%
3904   \acrfullformat{\glsentrylong{#1}}{\acronymfont{\glsentryshort{#1}}}%
3905 }
```

`\Glsentryfull`

```
3906 \newrobustcmd*\Glsentryfull}[1]{%
3907   \acrfullformat{\Glsentrylong{#1}}{\acronymfont{\glsentryshort{#1}}}%
3908 }
```

`\glsentryfullpl`

```
3909 \newcommand*\glsentryfullpl}[1]{%
3910   \acrfullformat{\glsentrylongpl{#1}}{\acronymfont{\glsentryshortpl{#1}}}%
3911 }
```

`\Glsentryfullpl`

```
3912 \newrobustcmd*\Glsentryfullpl}[1]{%
3913   \acrfullformat{\Glsentrylongpl{#1}}{\acronymfont{\glsentryshortpl{#1}}}%
3914 }
```

`\glsentrynumberlist` Displays the number list as is.

```
3915 \newcommand*\glsentrynumberlist}[1]{%
3916   \glsdoifexists{#1}%
3917   {%
3918     \@gls@entry@field{#1}{numberlist}%
3919   }%
3920 }
```

`\glsdisplaynumberlist` Formats the number list for the given entry label. Doesn't work with hyperref.

```
3921 \@ifpackageloaded{hyperref} {%
3922   \newcommand*\glsdisplaynumberlist}[1]{%
3923     \GlossariesWarning
3924     {%
3925       \string\glsdisplaynumberlist\space
3926       doesn't work with hyperref.^^JUsing
3927       \string\glsentrynumberlist\space instead%
3928     }%
3929     \glsentrynumberlist{#1}%
3930   }%
3931 }%
3932 {%
3933   \newcommand*\glsdisplaynumberlist}[1]{%
3934     \glsdoifexists{#1}%
3935     {%
3936       \bgroup
```

```

3937 \edef\@glo@label{\glsdetoklabel{#1}}%
3938 \let\@org@glsglnumberformat\glsglnumberformat
3939 \def\glsglnumberformat##1{##1}%
3940 \protected@edef\the@numberlist{%
3941   \csname glo@\@glo@label @numberlist\endcsname}%
3942 \def\@glsgl@numlist@sep{}%
3943 \def\@glsgl@numlist@nextsep{}%
3944 \def\@glsgl@numlist@lastsep{}%
3945 \def\@glsgl@thislist{}%
3946 \def\@glsgl@donext@def{}%
3947 \renewcommand\do[1]{%
3948   \protected@edef\@glsgl@thislist{%
3949     \@glsgl@thislist
3950     \noexpand\@glsgl@numlist@sep
3951     ##1%
3952   }%
3953   \let\@glsgl@numlist@sep\@glsgl@numlist@nextsep
3954   \def\@glsgl@numlist@nextsep{\glsgl@numlist@sep}%
3955   \@glsgl@donext@def
3956   \def\@glsgl@donext@def{%
3957     \def\@glsgl@numlist@lastsep{\glsgl@numlist@lastsep}%
3958   }%
3959 }%
3960 \expandafter \glsgl@numlist@parser \expandafter{\the@numberlist}%
3961 \let\@glsgl@numlist@sep\@glsgl@numlist@lastsep
3962 \@glsgl@thislist
3963 \egroup
3964 }%
3965 }
3966 }

```

`\glsgl@numlist@sep`

```

3967 \newcommand*{\glsgl@numlist@sep}{, }

```

`\glsgl@numlist@lastsep`

```

3968 \newcommand*{\glsgl@numlist@lastsep}{ \& }

```

`\glshyperlink` Provide a hyperlink to a glossary entry without adding information to the glossary file. The entry needs to be added using a command like `\glslink` or `\glsadd` to ensure that the target is defined. The first (optional) argument specifies the link text. The entry name is used by default. The second argument is the entry label.

```

3969 \newcommand*{\glshyperlink}[2][\glsgl@entrytext{\@glo@label}]{%
3970   \def\@glo@label{#2}%
3971   \@glslink{\glsgl@linkprefix\glsdetoklabel{#2}}{#1}}

```

1.11 Adding an entry to the glossary without generating text

The following keys are provided for `\glsadd` and `\glsaddall`:

```
3972 \define@key{glossadd}{counter}{\def\@gls@counter{#1}}
```

```
3973 \define@key{glossadd}{format}{\def\@glsnumberformat{#1}}
```

This key is only used by `\glsaddall`:

```
3974 \define@key{glossadd}{types}{\def\@glo@type{#1}}
```

`\glsadd[<options>]{<label>}`

Add a term to the glossary without generating any link text. The optional argument indicates which counter to use, and how to format it (using a key-value list) the second argument is the entry label. Note that *<options>* only has two keys: counter and format (the types key will be ignored).

`\glsadd`

```
3975 \newrobustcmd*{\glsadd}[2][]{%
3976   \glsdoifexists{#2}%
3977   {%
3978     \def\@glsnumberformat{glsnumberformat}%
3979     \edef\@gls@counter{\csname glo@glsetoklabel{#2}@counter\endcsname}%
3980     \setkeys{glossadd}{#1}%
```

Store the entry's counter in `\theglentrycounter`

```
3981   \@gls@saveentrycounter
3982   \@do@wrglossary{#2}%
3983 }%
3984 }
```

`\glsaddall[<option list>]`

Add all terms defined for the listed glossaries (without displaying any text). If types key is omitted, apply to all glossary types.

`\glsaddall`

```
3985 \newrobustcmd*{\glsaddall}[1][]{%
3986   \edef\@glo@type{\@glo@types}%
3987   \setkeys{glossadd}{#1}%
3988   \forallglsentries[\@glo@type]{\@glo@entry}{%
3989     \glsadd[#1]{\@glo@entry}%
3990   }%
3991 }
```

`\glsaddallunused`

`\glsaddallunused[<glossary type>]`

Add all used terms defined for the listed glossaries (without displaying any text). If optional argument is omitted, apply to all glossary types. This should typically go at the end of the document.

```

3992 \newrobustcmd*{\glsaddallunused}[1][\@glo@types]{%
3993 \forallglsentries[#1]{\@glo@entry}%
3994 {%
3995 \ifglsused{\@glo@entry}{\glsadd[format=@gobble]{\@glo@entry}}%
3996 }%
3997 }

```

1.12 Creating associated files

The `\writeist` command creates the associated customized `.ist` `makeindex` style file. While defining this command, some characters have their catcodes temporarily changed to ensure they get written to the `.ist` file correctly. The `makeindex` actual character (usually `@`) is redefined to be a `?`, to allow internal commands to be written to the glossary file output file.

The special characters are stored in `\@gls@actualchar`, `\@gls@encapchar`, `\@gls@levelchar` and `\@gls@quotechar` to make them easier to use later, but don't change these values, because the characters are encoded in the command definitions that are used to escape the special characters (which means that the user no longer needs to worry about `makeindex` special characters).

The symbols and numbers label for group headings are hardwired into the `.ist` file as `glssymbols` and `glsnumbers`, the group titles can be translated (so that `\glssymbolsgroupname` replaces `glssymbols` and `\glsnumbersgroupname` replaces `glsnumbers`) using the command `\glsgetgrouptitle` which is defined in . This is done to prevent any problem characters in `\glssymbolsgroupname` and `\glsnumbersgroupname` from breaking hyperlinks.

`\glsopenbrace` Define `\glsopenbrace` to make it easier to write an opening brace to a file.

```
3998 \edef\glsopenbrace{\expandafter\@gobble\string\{}
```

`\glsclosebrace` Define `\glsclosebrace` to make it easier to write an opening brace to a file.

```
3999 \edef\glsclosebrace{\expandafter\@gobble\string\}}
```

`\glsquote` Define command that makes it easier to write quote marks to a file in the event that the double quote character has been made active.

```
4000 \edef\glsquote#1{\string"#1\string"}
```

`\@glsfirstletter` Define the first letter to come after the digits 0,...,9. Only required for `xindy`.

```

4001 \ifglsxindy
4002 \newcommand*{\@glsfirstletter}{A}
4003 \fi

```

`stLetterAfterDigits` Sets the first letter to come after the digits 0,...,9.

```

4004 \ifglsxindy
4005 \newcommand*{\GlsSetXdyFirstLetterAfterDigits}[1]{%
4006 \renewcommand*{\@glsfirstletter}{#1}}
4007 \else

```

```

4008 \newcommand*{\GlsSetXdyFirstLetterAfterDigits}[1]{%
4009 \glsnoxindywarning\GlsSetXdyFirstLetterAfterDigits}
4010 \fi

\@glsminrange Define the minimum number of successive location references to merge into a
range.
4011 \newcommand*{\@glsminrange}{2}

etXdyMinRangeLength Set the minimum range length. The value must either be none or a positive
integer. The glossaries package doesn't check if the argument is valid, that is left
to xindy.
4012 \ifglxindy
4013 \newcommand*{\GlsSetXdyMinRangeLength}[1]{%
4014 \renewcommand*{\@glsminrange}{#1}}
4015 \else
4016 \newcommand*{\GlsSetXdyMinRangeLength}[1]{%
4017 \glsnoxindywarning\GlsSetXdyMinRangeLength}
4018 \fi

\writeist
4019 \ifglxindy
Code to use if xindy is required.
4020 \def\writeist{%
Define write register if not already defined
4021 \ifundef{\glswrite}{\newwrite\glswrite}{}%
Update attributes list
4022 \@gls@addpredefinedattributes
Open the file.
4023 \openout\glswrite=\istfilename
Write header comment at the start of the file
4024 \write\glswrite{;; xindy style file created by the glossaries
4025 package}%
4026 \write\glswrite{;; for document '\jobname' on
4027 \the\year-\the\month-\the\day}%
Specify the required styles
4028 \write\glswrite{^^J; required styles^^J}
4029 \@for\@xdystyle:=\@xdyrequiredstyles\do{%
4030 \ifx\@xdystyle\@empty
4031 \else
4032 \protected@write\glswrite{{(require
4033 \string"\@xdystyle.xdy\string")}}%
4034 \fi
4035 }%

```

List the allowed attributes (possible values used by the format key)

```
4036 \write\glswrite{^^J%
4037 ; list of allowed attributes (number formats)^^J}%
4038 \write\glswrite{(define-attributes ((\@xdyattributes)))}%
```

Define any additional alphabets

```
4039 \write\glswrite{^^J; user defined alphabets^^J}%
4040 \write\glswrite{\@xdyuseralphabets}%
```

Define location classes.

```
4041 \write\glswrite{^^J; location class definitions^^J}%
```

As from version 3.0, locations are now specified as $\{\langle Hprefix \rangle\}\{\langle number \rangle\}$, so need to add all possible combinations of location types.

```
4042 \@for\@gls@classI:=\@gls@xdy@locationlist\do{%
```

Case were $\langle Hprefix \rangle$ is empty:

```
4043 \protected@write\glswrite{}{(define-location-class
4044 \string"\@gls@classI\string"^^J\space\space\space
4045 (
4046 :sep "{{"
4047 \csname @gls@xdy@Lclass@\@gls@classI\endcsname\space
4048 :sep "}"
4049 )
4050 ^^J\space\space\space
4051 :min-range-length \@glsminrange^^J%
4052 )
4053 }%
```

Nested iteration over all classes:

```
4054 {%
4055 \@for\@gls@classII:=\@gls@xdy@locationlist\do{%
4056 \protected@write\glswrite{}{(define-location-class
4057 \string"\@gls@classII-\@gls@classI\string"
4058 ^^J\space\space\space
4059 (
4060 :sep "{"
4061 \csname @gls@xdy@Lclass@\@gls@classII\endcsname\space
4062 :sep "{{"
4063 \csname @gls@xdy@Lclass@\@gls@classI\endcsname\space
4064 :sep "}"
4065 )
4066 ^^J\space\space\space
4067 :min-range-length \@glsminrange^^J%
4068 )
4069 }%
4070 }%
4071 }%
4072 }%
```

User defined location classes (needs checking for new location format).

```

4073 \write\glswrite{^^J; user defined location classes}%
4074 \write\glswrite{\@xdyuserlocationdefs}%

```

Cross-reference class. (The unverified option is used as the cross-references are supplied using the list of labels along with the optional argument for `\glsseeformat` which xindy won't recognise.)

```

4075 \write\glswrite{^^J; define cross-reference class^^J}%
4076 \write\glswrite{(define-crossref-class \string"see\string"
4077 :unverified )}%

```

Define how cross-references should be displayed. This adds an empty set of braces after the cross-referencing information allowing for the final argument of `\glsseeformat` which gets ignored. (When using `makeindex` this final argument contains the location information which is not required.)

```

4078 \write\glswrite{(markup-crossref-list
4079 :class \string"see\string"^^J\space\space\space
4080 :open \string"\string\glsseeformat\string"
4081 :close \string"{}\string")}%

```

List the order to sort the classes.

```

4082 \write\glswrite{^^J; define the order of the location classes}%
4083 \write\glswrite{(define-location-class-order
4084 (\@xdylocationclassorder))}%

```

Specify what to write to the start and end of the glossary file.

```

4085 \write\glswrite{^^J; define the glossary markup^^J}%

4086 \write\glswrite{(markup-index^^J\space\space\space
4087 :open \string"\string
4088 \glossarysection[\string\glossarytoctitle]{\string
4089 \glossarytitle}\string\glossarypreamble}%

```

Add all the xindy-only macro definitions (needed to prevent errors in the event that the user changes from xindy to `makeindex`)

```

4090 \@for\@this@ctr:=\@xdycounters\do{%
4091   {%
4092     \@for\@this@attr:=\@xdyattributelist\do{%
4093       \protected\write\glswrite{}{\string\providecommand*%
4094         \expandafter\string
4095         \csname glsX\@this@ctr X\@this@attr\endcsname[2]%
4096         {%
4097           \string\setentrycounter
4098           [\expandafter\@gobble\string\#1]{\@this@ctr}%
4099           \expandafter\string
4100           \csname\@this@attr\endcsname
4101           {\expandafter\@gobble\string\#2}%
4102         }%
4103       }%
4104     }%
4105   }%
4106 }%

```


Add the end part of the open tag and the rest of the markup-index information:

```

4107 \write\glswrite{%
4108     \string\begin
4109     {theglossary}\string\glossaryheader\string~n\string" ^^J\space
4110     \space\space:close \string"\expandafter\@gobble
4111     \string~n\string~n\string
4112     \end{theglossary}\string\glossarypostamble
4113     \string~n\string" ^^J\space\space\space
4114     :tree)}}%

```

Specify what to put between letter groups

```

4115 \write\glswrite{(markup-letter-group-list
4116     :sep \string"\string\glsgroupskip\string~n\string"))}%

```

Specify what to put between entries

```

4117 \write\glswrite{(markup-indexentry
4118     :open \string"\string\relax \string\glsresetentrylist
4119     \string~n\string"))}%

```

Specify how to format entries

```

4120 \write\glswrite{(markup-locclass-list :open
4121     \string"\glsopenbrace\string\glossaryentrynumbers
4122     \glsopenbrace\string\relax\space \string"^^J\space\space\space
4123     :sep \string", \string"
4124     :close \string"\glsclosebrace\glsclosebrace\string"))}%

```

Specify how to separate location numbers

```

4125 \write\glswrite{(markup-locref-list
4126     :sep \string"\string\delimN\space\string"))}%

```

Specify how to indicate location ranges

```

4127 \write\glswrite{(markup-range
4128     :sep \string"\string\delimR\space\string"))}%

```

Specify 2-page and 3-page suffixes, if defined. First, the values must be sanitized to write them explicitly.

```

4129 \@onelevel@sanitize\gls@suffixF
4130 \@onelevel@sanitize\gls@suffixFF
4131 \ifx\gls@suffixF\@empty
4132 \else
4133     \write\glswrite{(markup-range
4134         :close "\gls@suffixF" :length 1 :ignore-end)}}%
4135 \fi
4136 \ifx\gls@suffixFF\@empty
4137 \else
4138     \write\glswrite{(markup-range
4139         :close "\gls@suffixFF" :length 2 :ignore-end)}}%
4140 \fi

```

Specify how to format locations.

```

4141 \write\glswrite{^^J; define format to use for locations^^J}%
4142 \write\glswrite{\@xdylocref}}%

```

Specify how to separate letter groups.

```
4143 \write\glswrite{^^J; define letter group list format^^J}%
4144 \write\glswrite{(markup-letter-group-list
4145 :sep \string"\string\glsgroupskip\string~n\string")}%
```

Define letter group headings.

```
4146 \write\glswrite{^^J; letter group headings^^J}%
4147 \write\glswrite{(markup-letter-group
4148 :open-head \string"\string\glsgroupheading
4149 \glsoopenbrace\string"^^J\space\space\space
4150 :close-head \string"\glsclosebrace\string")}%
```

Define additional letter groups.

```
4151 \write\glswrite{^^J; additional letter groups^^J}%
4152 \write\glswrite{\@xdylettergroups}%
```

Define additional sort rules

```
4153 \write\glswrite{^^J; additional sort rules^^J}
4154 \write\glswrite{\@xdysortrules}%
```

Close the style file

```
4155 \closeout\glswrite
```

Suppress any further calls.

```
4156 \let\writeist\relax
4157 }
4158 \else
```

Code to use if makeindex is required.

```
4159 \edef\@gls@actualchar{\string?}
4160 \edef\@gls@encapchar{\string|}
4161 \edef\@gls@levelchar{\string!}
4162 \edef\@gls@quotechar{\string"}
4163 \def\writeist{\relax
4164 \ifundef\glswrite{\newwrite\glswrite}{\relax
4165 \openout\glswrite=\istfilename
4166 \write\glswrite{\expandafter\@gobble\string}% makeindex style file
4167 created by the glossaries package}
4168 \write\glswrite{\expandafter\@gobble\string}% for document
4169 '\jobname' on \the\year-\the\month-\the\day}
4170 \write\glswrite{actual '\@gls@actualchar'}
4171 \write\glswrite{encap '\@gls@encapchar'}
4172 \write\glswrite{level '\@gls@levelchar'}
4173 \write\glswrite{quote '\@gls@quotechar'}
4174 \write\glswrite{keyword \string"\string\glossaryentry\string"}
4175 \write\glswrite{preamble \string"\string\glossarysection[\string
4176 \glossarytoctitle]{\string\glossarytitle}\string
4177 \glossarypreamble\string\n\string\begin{theglossary}\string
4178 \glossaryheader\string\n\string"}
4179 \write\glswrite{postamble \string"\string%\string\n\string
4180 \end{theglossary}\string\glossarypostamble\string\n
4181 \string"}
```

```

4182 \write\glswrite{group_skip \string"\string\glsgroupskip\string\n
4183 \string"}
4184 \write\glswrite{item_0 \string"\string%\string\n\string"}
4185 \write\glswrite{item_1 \string"\string%\string\n\string"}
4186 \write\glswrite{item_2 \string"\string%\string\n\string"}
4187 \write\glswrite{item_01 \string"\string%\string\n\string"}
4188 \write\glswrite{item_x1
4189 \string"\string\relax \string\glresetentrylist\string\n
4190 \string"}
4191 \write\glswrite{item_12 \string"\string%\string\n\string"}
4192 \write\glswrite{item_x2
4193 \string"\string\relax \string\glresetentrylist\string\n
4194 \string"}

4195 \write\glswrite{delim_0 \string"\string\{\string
4196 \glssymbols\string\{\string\relax \string"}
4197 \write\glswrite{delim_1 \string"\string\{\string
4198 \glssymbols\string\{\string\relax \string"}
4199 \write\glswrite{delim_2 \string"\string\{\string
4200 \glssymbols\string\{\string\relax \string"}
4201 \write\glswrite{delim_t \string"\string\}\string\}\string"}
4202 \write\glswrite{delim_n \string"\string\delimN \string"}
4203 \write\glswrite{delim_r \string"\string\delimR \string"}
4204 \write\glswrite{headings_flag 1}
4205 \write\glswrite{heading_prefix
4206 \string"\string\glsgroupheading\string\{\string"}
4207 \write\glswrite{heading_suffix
4208 \string"\string\}\string\relax
4209 \string\glresetentrylist \string"}
4210 \write\glswrite{symhead_positive \string"glssymbols\string"}
4211 \write\glswrite{numhead_positive \string"glssymbols\string"}
4212 \write\glswrite{page_compositor \string"glscpositor\string"}
4213 \@gls@escbsdq\gls@suffixF
4214 \@gls@escbsdq\gls@suffixFF
4215 \ifx\gls@suffixF\@empty
4216 \else
4217 \write\glswrite{suffix_2p \string"\gls@suffixF\string"}
4218 \fi
4219 \ifx\gls@suffixFF\@empty
4220 \else
4221 \write\glswrite{suffix_3p \string"\gls@suffixFF\string"}
4222 \fi
4223 \closeout\glswrite
4224 \let\writeist\relax
4225 }
4226 \fi

```

The command `\noist` will suppress the creation of the `.ist` file. Obviously you need to use this command before `\writeist` to have any effect.

`\noist`

```
4227 \newcommand{\noist}{%
```

Update attributes list

```
4228 \@gls@addpredefinedattributes
```

```
4229 \let\writeist\relax
```

```
4230 }
```

`\@makeglossary` is an internal command that takes an argument indicating the glossary type. This command will create the glossary file required by `makeindex` for the given glossary type, using the extension supplied by the `<out-ext>` parameter used in `\newglossary` (and it will also activate the `\glossary` command, and create the customized `.ist` `makeindex` style file).

Note that you can't use `\@makeglossary` for only some of the defined glossaries. You either need to have a `\makeglossary` for all glossaries or none (otherwise you will end up with a situation where \TeX is trying to write to a non-existent file). The relevant glossary must be defined prior to using `\@makeglossary`.

`\@makeglossary`

```
4231 \newcommand*{\@makeglossary}[1]{%
```

```
4232 \ifglossaryexists{#1}%
```

```
4233 {%
```

Only create a new write if `savewrites=false` otherwise create a token to collect the information.

```
4234 \ifglssavewrites
```

```
4235 \expandafter\newtoks\csname glo@#1@filetok\endcsname
```

```
4236 \else
```

```
4237 \expandafter\newwrite\csname glo@#1@file\endcsname
```

```
4238 \expandafter\@glsopenfile\csname glo@#1@file\endcsname{#1}%
```

```
4239 \fi
```

```
4240 \@gls@renewglossary
```

```
4241 \writeist
```

```
4242 }%
```

```
4243 {%
```

```
4244 \PackageError{glossaries}%
```

```
4245 {Glossary type ‘#1’ not defined}%
```

```
4246 {New glossaries must be defined before using \string\makeglossary}%
```

```
4247 }%
```

```
4248 }
```

`\@glsopenfile` Open write file associated with the given glossary.

```
4249 \newcommand*{\@glsopenfile}[2]{%
```

```
4250 \immediate\openout#1=\jobname.\csname @glotype@#2@out\endcsname
```

```
4251 \PackageInfo{glossaries}{Writing glossary file
```

```
4252 \jobname.\csname @glotype@#2@out\endcsname}%
```

```
4253 }
```

\@closegls

```
4254 \newcommand*{\@closegls}[1]{%
4255   \closeout\csname glo@#1@file\endcsname
4256 }
4257 %   \end{macrocode}
4258 %\end{macro}
4259 %
4260 %\begin{macro}{\@gls@automake}
4261 %\changes{4.08}{2014-07-30}{new}
4262 %   \begin{macrocode}
4263 \ifglxindy
4264   \newcommand*{\@gls@automake}[1]{%
4265     \ifglossaryexists{#1}
4266     {%
4267       \@closegls{#1}%
4268       \ifdefstring{glsorder}{letter}%
4269         {\def\@gls@order{-M ord/letorder }}%
4270         {\let\@gls@order\@empty}%
4271       \ifcsundef{xdy@#1@language}%
4272         {\let\@gls@langmod\@xdy@main@language}%
4273         {\letcs\@gls@langmod{xdy@#1@language}}%
4274       \edef\@gls@dothiswrite{\noexpand\write18{xindy
4275         -I xindy
4276         \@gls@order
4277         -L \@gls@langmod\space
4278         -M \@gls@istfilebase\space
4279         -C \@gls@codepage\space
4280         -t \jobname.\csuse{@glotype@#1@log}
4281         -o \jobname.\csuse{@glotype@#1@in}
4282         \jobname.\csuse{@glotype@#1@out}}}%
4283     }%
4284     \@gls@dothiswrite
4285   }%
4286   {%
4287     \GlossariesWarning{Can't make glossary '#1', it doesn't exist}%
4288   }%
4289 }
4290 \else
4291   \newcommand*{\@gls@automake}[1]{%
4292     \ifglossaryexists{#1}
4293     {%
4294       \@closegls{#1}%
4295       \ifdefstring{glsorder}{letter}%
4296         {\def\@gls@order{-l }}%
4297         {\let\@gls@order\@empty}%
4298       \edef\@gls@dothiswrite{\noexpand\write18{makeindex \@gls@order
4299         -s \istfilename\space
4300         -t \jobname.\csuse{@glotype@#1@log}
4301         -o \jobname.\csuse{@glotype@#1@in}}
```

```

4302      \jobname.\csuse{@glotype@#1@out}}%
4303    }%
4304    \@gls@dothiswrite
4305  }%
4306  {%
4307    \GlossariesWarning{Can't make glossary '#1', it doesn't exist}%
4308  }%
4309 }
4310 \fi

```

`\nomakeglossaries` Issue warning that `\makeglossaries` hasn't been used.

```

4311 \newcommand*{\@warn@nomakeglossaries}{%

```

Only use this if warning if `\printglossary` has been used without `\makeglossaries`

```

4312 \newcommand*{\warn@nomakeglossaries}{\@warn@nomakeglossaries}

```

`\makeglossaries` will use `\makeglossary` for each glossary type that has been defined. New glossaries need to be defined before using `\makeglossary`, so have `\makeglossaries` redefine `\newglossary` to prevent it being used afterwards.

`\makeglossaries`

```

4313 \newcommand*{\makeglossaries}{%

```

Define the write used for style file also used for all other output files if `savewrites=true`.

```

4314 \ifundef{glswrite}{\newwrite\glswrite}{}%

```

If the user removes the glossary package from their document, ensure the next run doesn't throw a load of undefined control sequence errors when the aux file is parsed.

```

4315 \protected@write\@auxout{}{\string\providecommand\string\@glsorder[1]{}%

```

```

4316 \protected@write\@auxout{}{\string\providecommand\string\@istfilename[1]{}%

```

Write the name of the style file to the aux file (needed by `makeglossaries`)

```

4317 \protected@write\@auxout{}{\string\@istfilename{\istfilename}}%

```

```

4318 \protected@write\@auxout{}{\string\@glsorder{\glsorder}}

```

Iterate through each glossary type and activate it.

```

4319 \@for\@glo@type:=\@glo@types\do{%

```

```

4320   \ifthenelse{\equal{\@glo@type}{}}{}{}%

```

```

4321   \@makeglossary{\@glo@type}}%

```

```

4322 }%

```

New glossaries must be created before `\makeglossaries` so disable `\newglossary`.

```

4323 \renewcommand*\newglossary[4][]{%

```

```

4324 \PackageError{glossaries}{New glossaries

```

```

4325 must be created before \string\makeglossaries}{You need

```

```

4326 to move \string\makeglossaries\space after all your

```

```

4327 \string\newglossary\space commands}}%

```

Any subsequence instances of this command should have no effect

```
4328 \let\@makeglossary\relax
4329 \let\makeglossary\relax
4330 \let\makeglossaries\relax
```

Disable all commands that have no effect after \makeglossaries

```
4331 \@disable@onlypremakeg
```

Allow see key:

```
4332 \let\gls@checkseeallowed\relax
```

Suppress warning about no \makeglossaries

```
4333 \let\warn@nomakeglossaries\relax
```

Activate warning about missing \printglossary

```
4334 \def\warn@noprntglossary{%
4335   \GlossariesWarningNoLine{No \string\printglossary\space
4336     or \string\printglossaries\space
4337     found.^^J(Remove \string\makeglossaries\space if you don't want
4338     any glossaries.)^^JThis document will not have a glossary}%
4339 }%
```

Declare list parser for \glsdisplaynumberlist

```
4340 \ifglssavenumberlist
4341   \edef\@gls@doddeflistparser{\noexpand\DeclareListParser
4342     {\noexpand\glsnumlistparser}{\delimN}}}%
4343   \@gls@doddeflistparser
4344 \fi
```

Prevent user from also using \makenoidxglossaries

```
4345 \let\makenoidxglossaries\@no@makeglossaries
```

Prohibit sort key in printgloss family:

```
4346 \renewcommand*{\@printgloss@setsort}{}%
4347 \let\@glo@assign@sortkey\@glo@no@assign@sortkey
4348 }%
```

Check the automake setting:

```
4349 \ifglsautomake
4350   \renewcommand*{\@gls@doautomake}{%
4351     \@for\@gls@type:=\@glo@types\do{%
4352       \ifdefempty{\@gls@type}{}%
4353       {\@gls@automake{\@gls@type}}}%
4354   }%
4355 }%
4356 \fi
4357 }
```

Must occur in the preamble:

```
4358 \@onlypreamble{\makeglossaries}
```

\glswrite The definition of \glswrite has now been moved to \makeglossaries so that it's only defined if needed.

The `\makeglossary` command is redefined to be identical to `\makeglossaries`.
(This is done to reinforce the message that you must either use `\@makeglossary`
for all the glossaries or for none of them.)

`\makeglossary`

```
4359 \let\makeglossary\makeglossaries
```

If `\makeglossaries` hasn't been used, issue a warning. Also issue a warning
if neither `\printglossaries` nor `\printglossary` have been used.

```
4360 \AtEndDocument{%
4361   \warn@nomakeglossaries
4362   \warn@noprintglossary
4363 }
```

`\makenoidxglossaries` Analogous to `\makeglossaries` this activates the commands needed for `\printnoidxglossary`

```
4364 \newcommand*\makenoidxglossaries{%
```

Redefine empty glossary warning:

```
4365   \renewcommand{\@gls@noref@warn}[1]{%
4366     \GlossariesWarning{Empty glossary for
4367       \string\printnoidxglossary[type={##1}].
4368     Rerun may be required (or you may have forgotten to use
4369     commands like \string\gls).}%
4370 }
```

Don't escape makeindex/xindy characters

```
4371 \let\@gls@checkmkidxchars\@gobble
```

Write glossary information to aux instead of glossary files

```
4372 \let\@do@wrglossary\gls@noidxglossary
```

Switch on group headings that use the character code:

```
4373 \let\@gls@getgrouptitle\@gls@noidx@getgrouptitle
```

Allow see key:

```
4374 \let\gls@checkseeallowed\relax
```

Redefine cross-referencing macro:

```
4375 \renewcommand{\@do@seeglossary}[2]{%
4376   \edef\@gls@label{\glsdetoklabel{##1}}%
4377   \protected@write\@auxout{%
4378     \string\@gls@reference
4379     {\csname glo@\@gls@label @type\endcsname}%
4380     {\@gls@label}%
4381     {%
4382       \string\glsseeformat##2}%
4383     }%
4384   }%
4385 }
```


If user removes the glossaries package from their document, ensure the next run doesn't throw a load of undefined control sequence errors when the aux file is parsed.

```
4386 \AtBeginDocument
4387 {%
4388 \write\@auxout{\string\providecommand\string\@gls@reference[3]{}}%
4389 }%
```

Change warning about no glossares

```
4390 \def\warn@noprintglossary{%
4391 \GlossariesWarningNoLine{No \string\printnoidxglossary\space
4392 or \string\printnoidxglossaries ^^J
4393 found. (Remove \string\makenoidxglossaries\space if you
4394 don't want any glossaries.)^^JThis document will not have a glossary}%
4395 }%
```

Suppress warning about no \makeglossaries

```
4396 \let\warn@nomakeglossaries\relax
```

Prevent user from also using \makeglossaries

```
4397 \let\makeglossaries\@no@makeglossaries
```

Allow sort key in printgloss family:

```
4398 \renewcommand*{\@printgloss@setsort}{%
4399 \let\@glo@assign@sortkey\@glo@assign@sortkey
```

Initialise default sort order:

```
4400 \def\@glo@sorttype{\@glo@default@sorttype}%
4401 }%
```

All entries must be defined in the preamble:

```
4402 \renewcommand*\new@glossaryentry[2]{%
4403 \PackageError{glossaries}{Glossary entries must be
4404 defined in the preamble^^Jwhen you use
4405 \string\makenoidxglossaries}%
4406 {Either move your definitions to the preamble or use
4407 \string\makeglossaries}%
4408 }%
```

Redefine \glsentrynumberlist

```
4409 \renewcommand*{\glsentrynumberlist}[1]{%
4410 \letcs{\@gls@loclist}{\glo@\glsdetoklabel{##1}@loclist}%
4411 \ifdef\@gls@loclist
4412 {%
4413 \glsnoidxloclist{\@gls@loclist}%
4414 }%
4415 {%
4416 \ifglsentryexists{##1}%
4417 {%
4418 \GlossariesWarning{Missing location list for ‘##1’. Either
4419 a rerun is required or you haven't referenced the entry.}%
4420 }%
```

```

4421     {%
4422         \PackageError{glossaries}{Glossary entry ‘##1’ has not been
4423             defined.}{}%
4424     }%
4425 }%
4426 }%

Redefine \glsdisplaynumberlist
4427 \renewcommand*{\glsdisplaynumberlist}[1]{%
4428     \letcs{\@gls@loclist}{glo@glstoklabel{##1}@loclist}%
4429     \ifdef\@gls@loclist
4430     {%
4431         \def\@gls@noidxloclist@sep{%
4432             \def\@gls@noidxloclist@sep{%
4433                 \def\@gls@noidxloclist@sep{%
4434                     \glsnumlistsep
4435                 }%
4436                 \def\@gls@noidxloclist@finalsep{\glsnumlistlastsep}%
4437             }%
4438         }%
4439         \def\@gls@noidxloclist@finalsep{}%
4440         \def\@gls@noidxloclist@prev{}%
4441         \forlistloop{\glsnoidxdisplayloclisthandler}{\@gls@loclist}%
4442         \@gls@noidxloclist@finalsep
4443         \@gls@noidxloclist@prev
4444     }%
4445     {%
4446         ??\ifglsentryexists{##1}%
4447         {%
4448             \GlossariesWarning{Missing location list for ‘##1’. Either
4449                 a rerun is required or you haven’t referenced the entry.}%
4450         }%
4451         {%
4452             \PackageError{glossaries}{Glossary entry ‘##1’ has not been
4453                 defined.}{}%
4454         }%
4455     }%
4456 }%

```

Provide a generic way of iterating through the number list:

```

4457 \renewcommand*{\glsnumberlistloop}[3]{%
4458     \letcs{\@gls@loclist}{glo@glstoklabel{##1}@loclist}%
4459     \let\@gls@org@glsnoidxdisplayloc\glsnoidxdisplayloc
4460     \let\@gls@org@glsseeformat\glsseeformat
4461     \let\glsnoidxdisplayloc##2\relax
4462     \let\glsseeformat##3\relax
4463     \ifdef\@gls@loclist
4464     {%
4465         \forlistloop{\glsnoidxnumberlistloophandler}{\@gls@loclist}%
4466     }%

```

```

4467   {%
4468     \ifglsentryexists{##1}%
4469     {%
4470       \GlossariesWarning{Missing location list for ‘##1’. Either
4471         a rerun is required or you haven’t referenced the entry.}%
4472     }%
4473     {%
4474       \PackageError{glossaries}{Glossary entry ‘##1’ has not been
4475         defined.}{}%
4476     }%
4477   }%
4478   \let\glsnoidxdisplayloc\@gls@org\glsnoidxdisplayloc
4479   \let\glsseeformat\@gls@org\glsseeformat
4480 }%

```

Modify sanitize sort function

```

4481 \let\@gls@sanitizesort\@gls@noidx@sanitizesort
4482 \let\@gls@nosanitizesort\@gls@noidx@nosanitizesort
4483 \@gls@noidx@setsanitizesort
4484 }

```

Preamble-only command:

```

4485 \onlypreamble{\makenoidxglossaries}

```

`\glsnumberlistloop` `\glsnumberlistloop{<label>}{<handler>}`

```

4486 \newcommand*{\glsnumberlistloop}[2]{%
4487   \PackageError{glossaries}{\string\glsnumberlistloop\space
4488     only works with \string\makenoidxglossaries}{}%
4489 }

```

`numberlistloophandler` Handler macro for `\glsnumberlistloop`. (The argument should be in the form `\glsnoidxdisplayloc{<prefix>}{<counter>}{<format>}{<n>}`)

```

4490 \newcommand*{\glsnoidxnumberlistloophandler}[1]{%
4491   #1%
4492 }

```

`\@no@makeglossaries` Can’t use both `\makeglossaries` and `\makenoidxglossaries`

```

4493 \newcommand*{\@no@makeglossaries}{%
4494   \PackageError{glossaries}{You can’t use both
4495     \string\makeglossaries\space and \string\makenoidxglossaries}%
4496   {Either use one or other (or none) of those commands but not both
4497     together.}%
4498 }

```

`\@gls@noref@warn` Warning when no instances of `\@gls@reference` found.

```

4499 \newcommand{\@gls@noref@warn}[1]{%
4500   \GlossariesWarning{\string\makenoidxglossaries\space

```

```

4501   is required to make \string\printnoidxglossary[type={#1}] work}%
4502 }

```

`\gls@noidxglossary` Write the glossary information to the aux file:

```

4503 \newcommand*{\gls@noidxglossary}{%
4504   \protected@write\@auxout{}{%
4505     \string\@gls@reference
4506     {\csname glo@\@gls@label @type\endcsname}%
4507     {\@gls@label}%
4508     {\string\glsnoidxdisplayloc
4509      {\@glo@counterprefix}%
4510      {\@gls@counter}%
4511      {\@glsnumberformat}%
4512      {\@glslocref}%
4513     }%
4514   }%
4515 }

```

1.13 Writing information to associated files

`\istfile` Deprecated.

```

4516 \def\istfile{\glswrite}

```

At the end of the document, the files should be created if `savewrites=true`.

```

4517 \AtEndDocument{%
4518   \glswritefiles
4519 }

```

`\@glswritefiles` Only write the files if `savewrites=true`

```

4520 \newcommand*{\@glswritefiles}{%

```

Iterate through all the glossaries

```

4521   \foralllglossaries{\@glo@type}{%

```

Check for empty glossaries (patch provided by Patrick Häcker)

```

4522     \ifcsundef{glo@\@glo@type @filetok}%
4523     {%
4524       \def\gls@tmp{}%
4525     }%
4526     {%
4527       \edef\gls@tmp{\expandafter\the
4528         \csname glo@\@glo@type @filetok\endcsname}%
4529     }%
4530     \ifx\gls@tmp\@empty
4531       \ifx\@glo@type\glsdefaulttype
4532         \GlossariesWarningNoLine{Glossary '\@glo@type' has no
4533           entries.^^JRemember to use package option 'nomain' if
4534 you
4535           don't want to^^Juse the main glossary}%

```

```

4536         \else
4537             \GlossariesWarningNoLine{Glossary ‘\@glo@type’ has no
4538                 entries}%
4539         \fi
4540     \else
4541         \@glsopenfile{\glswrite}{\@glo@type}%
4542         \immediate\write\glswrite{%
4543             \expandafter\the
4544             \csname glo@\@glo@type @filetok\endcsname}%
4545         \immediate\closeout\glswrite
4546     \fi
4547 }%
4548 }

```

The `\glossary` command is redefined so that it takes an optional argument *<type>* to specify the glossary type (use `\glsdefaulttype` glossary by default). This shouldn't be used at user level as `\glslink` sets the correct format. The associated number should be stored in `\theglsentrycounter` before using `\glossary`.

`\glossary`

```

4549 \renewcommand*{\glossary}[1][\glsdefaulttype]{%
4550     \@glossary[#1]%
4551 }

```

Define internal `\@glossary` to ignore its argument. This gets redefined in `\@makeglossary`. This is defined to just `\index` as memoir changes the definition of `\@index`. (Thanks to Dan Luecking for pointing this out.)

`\@glossary`

```

4552 \def\@glossary[#1]{\index}

```

This is a convenience command to set `\@glossary`. It is used by `\@makeglossary` and then redefined to do nothing, as it only needs to be done once.

`\@gls@renewglossary`

```

4553 \newcommand{\@gls@renewglossary}{%
4554     \gdef\@glossary[##1]{\@bsphack\begingroup\@wrglossary{##1}}%
4555     \let\@gls@renewglossary\@empty
4556 }

```

The `\@wrglossary` command is redefined to have two arguments. The first argument is the glossary type, the second argument is the glossary entry (the format of which is set in `\glslink`).

`\@wrglossary`

```

4557 \renewcommand*{\@wrglossary}[2]{%
4558     \ifglssavewrites
4559         \protected@edef\@gls@tmp{\the\csname glo@#1@filetok\endcsname#2}%

```

```

4560 \expandafter\global\expandafter\csname glo@#1@filetok\endcsname
4561 \expandafter{\@gls@tmp^^J}%
4562 \else

4563 \ifcsdef{glo@#1@file}%
4564 {%
4565 \expandafter\protected@write\csname glo@#1@file\endcsname{%
4566 \gls@disablepagerefexpansion}{#2}%
4567 }%
4568 {%
4569 \ifignoredglossary{#1}{}%
4570 {%
4571 \GlossariesWarning{No file defined for glossary ‘#1’}%
4572 }%
4573 }%
4574 \fi
4575 \endgroup\@esphack
4576 }

```

\do@wrglossary

```

4577 \newcommand*{\do@wrglossary}[1]{%
4578 \ifglsindexonlyfirst
4579 \ifglsused{#1}{\do@wrglossary{#1}}%
4580 \else
4581 \@do@wrglossary{#1}%
4582 \fi
4583 }

```

@protected@pagefmts List of page formats to be protected against expansion.

```

4584 \newcommand{\gls@protected@pagefmts}{%
4585 \gls@numberpage,\gls@alphpage,\gls@Alphpage,\gls@romanpage,\gls@Romanpage%
4586 }

```

blepagerefexpansion

```

4587 \newcommand*{\gls@disablepagerefexpansion}{%
4588 \@for\@gls@this:=\gls@protected@pagefmts\do
4589 {%
4590 \expandafter\let\@gls@this\relax
4591 }%
4592 }

```

\gls@alphpage

```

4593 \newcommand*{\gls@alphpage}{\@alph\c@page}

```

\gls@Alphpage

```

4594 \newcommand*{\gls@Alphpage}{\@Alph\c@page}

```

\gls@numberpage

```

4595 \newcommand*{\gls@numberpage}{\number\c@page}

```

```

\gls@romanpage
4596 \newcommand*{\gls@romanpage}{\romannumeral\c@page}

\gls@Romanpage
4597 \newcommand*{\gls@Romanpage}{\@Roman\c@page}

\@do@wrglossary Write the glossary entry in the appropriate format. (Need to set \glsnumberformat
and \gls@counter prior to use.) The argument is the entry's label.
4598 \newcommand*{\@do@wrglossary}[1]{%
4599 \begingroup

First a bit of hackery to prevent premature expansion of \c@page. Store original
definitions:
4600 \let\orgthe\the
4601 \let\orgnumber\number
4602 \let\orgromannumeral\romannumeral
4603 \let\orgalph\@alph
4604 \let\orgAlph\@Alph
4605 \let\orgRoman\@Roman

Redefine:
4606 \def\the##1{%
4607 \ifx##1\c@page \gls@numberpage\else\orgthe##1\fi}%
4608 \def\number##1{%
4609 \ifx##1\c@page \gls@numberpage\else\orgnumber##1\fi}%
4610 \def\romannumeral##1{%
4611 \ifx##1\c@page \gls@romanpage\else\orgromannumeral##1\fi}%
4612 \def\@Roman##1{%
4613 \ifx##1\c@page \gls@Romanpage\else\orgRoman##1\fi}%
4614 \def\@alph##1{%
4615 \ifx##1\c@page \gls@alphpage\else\orgalph##1\fi}%
4616 \def\@Alph##1{%
4617 \ifx##1\c@page \gls@Alphpage\else\orgAlph##1\fi}%

Prevent expansion:
4618 \gls@disablepagerefexpansion

Now store location in \glslocref:
4619 \protected@xdef\glslocref{\theHglentrycounter}%
4620 \endgroup

Escape any special characters
4621 \@gls@checkmkidxchars\glslocref

Check if the hyper-location is the same as the location and set the hyper prefix.

4622 \expandafter\ifx\theHglentrycounter\theHglentrycounter\relax
4623 \def\@glo@counterprefix{%
4624 \else
4625 \protected@edef\@glsHlocref{\theHglentrycounter}%
4626 \@gls@checkmkidxchars\@glsHlocref

```

```

4627 \edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix
4628 {\@gls@locref}{\@gls@Hlocref}}%
4629 }%
4630 \@do@gls@getcounterprefix
4631 \fi

```

De-tok label if required

```

4632 \edef\@gls@label{\glsdetoklabel{#1}}%

```

Write the information to file:

```

4633 \@do@wrglossary
4634 }

```

\@do@wrglossary

```

4635 \newcommand*{\@do@wrglossary}{%

```

Determine whether to use xindy or makeindex syntax

```

4636 \ifglsxindy

```

Need to determine if the formatting information starts with a (or) indicating a range.

```

4637 \expandafter\@glo@check@mkidxrangechar\@gls@numberformat\@nil
4638 \def\@glo@range{}%
4639 \expandafter\if\@glo@prefix(\relax
4640 \def\@glo@range{:open-range}%
4641 \else
4642 \expandafter\if\@glo@prefix)\relax
4643 \def\@glo@range{:close-range}%
4644 \fi
4645 \fi

```

Write to the glossary file using xindy syntax.

```

4646 \glossary[\csname glo@\@gls@label @type\endcsname]{%
4647 (indexentry :tkey (\csname glo@\@gls@label @index\endcsname)
4648 :locref \string"\@glo@counterprefix}{\@gls@locref}\string" %
4649 :attr \string"\@gls@counter\@glo@suffix\string"
4650 \@glo@range
4651 )
4652 }%
4653 \else

```

Convert the format information into the format required for makeindex

```

4654 \@set@glo@numformat{\@glo@numfmt}{\@gls@counter}{\@gls@numberformat}}%
4655 {\@glo@counterprefix}%

```

Write to the glossary file using makeindex syntax.

```

4656 \glossary[\csname glo@\@gls@label @type\endcsname]{%
4657 \string\glossaryentry{\csname glo@\@gls@label @index\endcsname
4658 \@gls@encapchar\@glo@numfmt}{\@gls@locref}}%
4659 \fi
4660 }

```


`\ls@getcounterprefix` Get the prefix that needs to be prepended to counter in order to get the hyper counter. (For example, with the standard article class and hyperref, `\theequation` needs to be prefixed with `\section num` to get the equivalent `\theHequation`.) NB this assumes that the prefix ends with a dot, which is the standard. (Otherwise it makes the xindy location classes more complicated.)

```

4661 \newcommand*\@gls@getcounterprefix[2]{%
4662   \edef\@gls@thisloc{#1}\edef\@gls@thisHloc{#2}%
4663   \ifx\@gls@thisloc\@gls@thisHloc
4664     \def\@glo@counterprefix{}%
4665   \else
4666     \def\@gls@get@counterprefix##1.#1##2\end@getprefix{%
4667       \def\@glo@tmp{##2}%
4668       \ifx\@glo@tmp\@empty
4669         \def\@glo@counterprefix{}%
4670       \else
4671         \def\@glo@counterprefix{##1}%
4672       \fi
4673     }%
4674   \@gls@get@counterprefix#2.#1\end@getprefix

```

Warn if no prefix can be formed.

```

4675   \ifx\@glo@counterprefix\@empty
4676     \GlossariesWarning{Hyper target ‘#2’ can’t be formed by
4677       prefixing~Jlocation ‘#1’. You need to modify the
4678       definition of \string\theH\@gls@counter~Jotherwise you
4679       will get the warning: “‘name{\@gls@counter.#1}’ has been~J
4680       referenced but does not exist”}%
4681   \fi
4682 \fi
4683 }

```

1.14 Glossary Entry Cross-References

`\@do@seeglossary` Write the glossary entry with a cross reference. The first argument is the entry’s label, the second must be in the form `[\langle tag \rangle]{\langle list \rangle}`, where `\langle tag \rangle` is a tag such as “see” and `\langle list \rangle` is a list of labels.

```

4684 \newcommand{\@do@seeglossary}[2]{%
4685   \def\@gls@xref{#2}%
4686   \@onelevel@sanitize\@gls@xref
4687   \@gls@checkmkidxchars\@gls@xref
4688   \ifglxsindy
4689     \glossary[\csname glo@#1@type\endcsname]{%
4690       (indexentry
4691         :tkey (\csname glo@#1@index\endcsname)
4692         :xref (\string"\@gls@xref\string")
4693         :attr \string"see\string"
4694       )
4695     }%

```

```

4696 \else
4697   \glossary[\csname glo@#1@type\endcsname]{%
4698   \string\glossaryentry{\csname glo@#1@index\endcsname
4699   \@gls@encapchar glsseeformat\@gls@xref}{Z}}}%
4700 \fi
4701 }

```

`\@gls@fixbraces` If no optional argument is specified, list needs to be enclosed in a set of braces.

```

4702 \def\@gls@fixbraces#1#2#3\@nil{%
4703   \ifx#2[\relax
4704     \@gls@fixbraces#1#2#3\@end@fixbraces
4705   \else
4706     \def#1{{#2#3}}}%
4707   \fi
4708 }

```

`\@@gls@fixbraces`

```

4709 \def\@@gls@fixbraces#1[#2]#3\@end@fixbraces{%
4710   \def#1{[#2]{#3}}}%
4711 }

```

`\glssee` `\glssee{<label>}{<cross-ref list>}`

```

4712 \DeclareRobustCommand*\glssee}[3][\seename]{%
4713   \@do@seeglossary{#2}{#1}{#3}}%
4714 \newcommand*\@glssee}[3][\seename]{%
4715   \glssee[#1]{#3}{#2}}

```

`\glsseeformat` The first argument specifies what tag to use (e.g. “see”), the second argument is a comma-separated list of labels. The final argument (the location) is ignored.

```

4716 \DeclareRobustCommand*\glsseeformat}[3][\seename]{%
4717   \emph{#1} \glsseelist{#2}}

```

`\glsseelist` `\glsseelist{<list>}` formats list of entry labels.

```

4718 \DeclareRobustCommand*\glsseelist}[1]{%

```

If there is only one item in the list, set the last separator to do nothing.

```

4719   \let\@gls@dolast\relax

```

Don’t display separator on the first iteration of the loop

```

4720   \let\@gls@donext\relax

```

Iterate through the labels

```

4721   \@for\@gls@thislabel:=#1\do{%

```

Check if on last iteration of loop

```

4722     \ifx\@xfor@nextelement\@nnil

```

```

4723       \@gls@dolast

```

```

4724     \else

```

```

4725       \@gls@donext

```

```

4726     \fi

```

Display the entry for this label. (Expanding label as it's a temporary control sequence that's used elsewhere.)

```
4727 \expandafter\glsseeitem\expandafter{\@gls@thislabel}%
```

Update separators

```
4728 \let\@gls@dolast\glsseelastsep
```

```
4729 \let\@gls@donext\glsseesep
```

```
4730 }%
```

```
4731 }
```

`\glsseelastsep` Separator to use between penultimate and ultimate entries in a cross-referencing list.

```
4732 \newcommand*{\glsseelastsep}{\space\andname\space}
```

`\glsseesep` Separator to use between entries in a cross-referencing list.

```
4733 \newcommand*{\glsseesep}{, }
```

`\glsseeitem` `\glsseeitem{<label>}` formats individual entry in a cross-referencing list.

```
4734 \DeclareRobustCommand*{\glsseeitem}[1]{\gls hyperlink[\glsseeitemformat{#1}]{#1}}
```

`\glsseeitemformat` As from v3.0, default is to use `\glsentrytext` instead of `\glsentryname`. (To avoid problems with the name key being sanitized.)

```
4735 \newcommand*{\glsseeitemformat}[1]{\glsentrytext{#1}}
```

1.15 Displaying the glossary

An individual glossary is displayed in the text using `\printglossary[<key-val list>]`. If the type key is omitted, the default glossary is displayed. The optional argument can be used to specify an alternative glossary, and can also be used to set the style, title and entry in the table of contents. Available keys are defined below.

`\gls@save@numberlist` Provide command to store number list.

```
4736 \newcommand*{\gls@save@numberlist}[1]{%
```

```
4737 \ifglssavenumberlist
```

```
4738 \toks@{#1}%
```

```
4739 \edef\@do@writeaux@info{%
```

```
4740 \noexpand\csgdef{glo@\glscurrententrylabel @numberlist}{\the\toks@}%
```

```
4741 }%
```

```
4742 \@onelevel@sanitize\@do@writeaux@info
```

```
4743 \protected@write\@auxout{}{\@do@writeaux@info}%
```

```
4744 \fi
```

```
4745 }
```

`\warn@noprintglossary` Warn the user if they have forgotten `\printglossaries` or `\printglossary`. (Will be suppressed if there is at least one occurrence of `\printglossary`.)

There is no check to ensure that there is a `\printglossary` for each defined glossary.)

```
4746 \newcommand*{\warn@noprintglossary}{}%
```

`\printglossary` The TOC title needs to be processed in a different manner to the main title in case the translator and hyperref packages are both being used.

```
4747 \ifcsundef{printglossary}{}%
```

```
4748 {%
```

If `\printglossary` is already defined, issue a warning and undefine it.

```
4749 \glswarnonglossdefined
```

```
4750 \undef\printglossary
```

```
4751 }
```

`\printglossary` has an optional argument. The default value is to set the glossary type to the main glossary.

```
4752 \newcommand*{\printglossary}[1][type=\glsdefaulttype]{%
```

```
4753 \@printglossary{#1}{\@print@glossary}%
```

```
4754 }
```

The `\printglossaries` command will do `\printglossary` for each glossary type that has been defined. It is better to use `\printglossaries` rather than individual `\printglossary` commands to ensure that you don't forget any new glossaries you may have created. It also makes it easier to chop and change the value of the acronym package option. However, if you want to list the glossaries in a different order, or if you want to set the title or table of contents entry, or if you want to use different glossary styles for each glossary, you will need to use `\printglossary` explicitly for each glossary type.

`\printglossaries`

```
4755 \newcommand*{\printglossaries}{%
```

```
4756 \foralllglossaries{\@glo@type}{\printglossary[type=\@glo@type]}%
```

```
4757 }
```

`\printnoidxglossary` Provide an alternative to `\printglossary` that doesn't require an external indexing application. Entries won't be sorted and the location list will be empty.

```
4758 \newcommand*{\printnoidxglossary}[1][type=\glsdefaulttype]{%
```

```
4759 \@printglossary{#1}{\@print@noidx@glossary}%
```

```
4760 }
```

`\printnoidxglossaries` Analogous to `\printglossaries`

```
4761 \newcommand*{\printnoidxglossaries}{%
```

```
4762 \foralllglossaries{\@glo@type}{\printnoidxglossary[type=\@glo@type]}%
```

```
4763 }
```

`@printgloss@setsort` Initialise to do nothing.

```
4764 \newcommand*{\@printgloss@setsort}{}
```

`\@printglossary` Sets up the glossary for either `\printglossary` or `\printnoidxglossary`. The first argument is the options list, the second argument is the handler macro that deals with the actual glossary.

```
4765 \newcommand{\@printglossary}[2]{%
```

Set up defaults.

```
4766 \def\@glo@type{\glsdefaulttype}%
4767 \def\glossarytitle{\csname @glo@type\@glo@type @title\endcsname}%

4768 \def\glossarytoctitle{\glossarytitle}%
4769 \let\org@glossarytitle\glossarytitle
4770 \def\@glossarystyle{}%
4771 \def\gls@dotoc@title{\glssettoc@title{\@glo@type}}%
```

Store current value of `\glossaryentrynumbers`. (This may be changed via the optional argument)

```
4772 \let\org@glossaryentrynumbers\glossaryentrynumbers
```

Localise the effects of the optional argument

```
4773 \bgroup
```

Activate or deactivate sort key:

```
4774 \@printgloss@setsort
```

Determine settings specified in the optional argument.

```
4775 \setkeys{printgloss}{#1}%
```

If title has been set, but toctitle hasn't, make toctitle the same as given title (rather than the title used when the glossary was defined)

```
4776 \ifx\glossarytitle\org@glossarytitle
4777 \else
4778 \expandafter\let\csname @glo@type\@glo@type @title\endcsname
4779 \glossarytitle
4780 \fi
```

Allow a high-level user command to indicate the current glossary

```
4781 \let\currentglossary\@glo@type
```

Enable individual number lists to be suppressed.

```
4782 \let\org@glossaryentrynumbers\glossaryentrynumbers
4783 \let\glsnonextpages\@glsnonextpages
```

Enable individual number list to be activated:

```
4784 \let\glsnextpages\@glsnextpages
```

Enable suppression of description terminators.

```
4785 \let\nopostdesc\@nopostdesc
```

Set up the entry for the TOC

```
4786 \gls@dotoc@title
```

Set the glossary style

```
4787 \@glossarystyle
```

Added a way to fetch the current entry label (v3.08 updated for new `\glossentry` and `\subglossentry`, but this is now only needed for backward compatibility):

```

4788 \let\gls@org@glossaryentryfield\glossentry
4789 \let\gls@org@glossarysubentryfield\subglossentry
4790 \renewcommand{\glossentry}[1]{%
4791   \xdef\glscurrententrylabel{\glsdetoklabel{##1}}%
4792   \gls@org@glossaryentryfield{##1}%
4793 }%
4794 \renewcommand{\subglossentry}[2]{%
4795   \xdef\glscurrententrylabel{\glsdetoklabel{##2}}%
4796   \gls@org@glossarysubentryfield{##1}{##2}%
4797 }%

```

Now do the handler macro that deals with the actual glossary:

```

4798   #2%

End the current scope
4799 \egroup

Reset \glossaryentrynumbers
4800 \global\let\glossaryentrynumbers\@org@glossaryentrynumbers

Suppress warning about no \printglossary
4801 \global\let\warn@noprintglossary\relax
4802 }

```

`\@print@glossary` Internal workings of `\printglossary` dealing with reading the external file.

```

4803 \newcommand{\@print@glossary}{%

Some macros may end up being expanded into internals in the glossary, so
need to make @ a letter. (Unlikely to be a problem since v3.08a but kept for
backward compatibility.)

4804 \makeatletter

Input the glossary file, if it exists.
4805 \@input@{\jobname.\csname @glotype@\@glo@type @in\endcsname}%

If the glossary file doesn't exist, do \null. (This ensures that the page is shipped
out and all write commands are done.) This might produce an empty page, but
at this point the document isn't complete, so it shouldn't matter.
4806 \IfFileExists{\jobname.\csname @glotype@\@glo@type @in\endcsname}%
4807 {}%
4808 {\null}%

If xindy is being used, need to write the language dependent information to
the .aux file for makeglossaries.
4809 \ifglxindy
4810   \ifcsundef{@xdy@\@glo@type @language}%
4811   {%
4812     \edef\@do@auxoutstuff{%
4813       \noexpand\AtEndDocument{%

```

If the user removes the glossary package from their document, ensure the next run doesn't throw a load of undefined control sequence errors when the aux file is parsed.

```

4814         \noexpand\immediate\noexpand\write\@auxout{%
4815         \string\providecommand\string\@xdylanguage[2]{}%
4816         \noexpand\immediate\noexpand\write\@auxout{%
4817         \string\@xdylanguage{\@glo@type}{\@xdy@main@language}}%
4818     }%
4819 }%
4820 }%
4821 {%
4822     \edef\@do@auxoutstuff{%
4823     \noexpand\AtEndDocument{%
4824     \noexpand\immediate\noexpand\write\@auxout{%
4825     \string\providecommand\string\@xdylanguage[2]{}%
4826     \noexpand\immediate\noexpand\write\@auxout{%
4827     \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type
4828     @language\endcsname}}%
4829     }%
4830     }%
4831     }%
4832     \@do@auxoutstuff
4833     \edef\@do@auxoutstuff{%
4834     \noexpand\AtEndDocument{%

```

If the user removes the glossaries package from their document, ensure the next run doesn't throw a load of undefined control sequence errors when the aux file is parsed.

```

4835         \noexpand\immediate\noexpand\write\@auxout{%
4836         \string\providecommand\string\@gls@codepage[2]{}%
4837         \noexpand\immediate\noexpand\write\@auxout{%
4838         \string\@gls@codepage{\@glo@type}{\gls@codepage}}%
4839     }%
4840 }%
4841 \@do@auxoutstuff
4842 \fi

```

Activate warning if \makeglossaries hasn't been used.

```

4843 \renewcommand*{\@warn@nomakeglossaries}{%
4844 \GlossariesWarningNoLine{\string\makeglossaries\space
4845 hasn't been used,^^Jthe glossaries will not be updated}%
4846 }%
4847 }

```

The sort macros all have the syntax:

`\@glo@sortmacro@<order>{<type>}`

where $\langle order \rangle$ is the sort order as specified by the sort key and $\langle type \rangle$ is the glossary type. (The referenced entry list is stored in $\backslash@glsref@{\langle type \rangle}$. The actual sorting is done by $\backslash@glo@sortentries{\langle handler \rangle}{\langle type \rangle}$).

$\backslash@glo@sortentries$

```

4848 \newcommand*{\@glo@sortentries}[2]{%
4849   \def\@glo@sortinglist{}%
4850   \def\@glo@sortinghandler{#1}%
4851   \edef\@glo@type{#2}%
4852   \forlistcsloop{\@glo@do@sortentries}{\@glsref@#2}%
4853   \csdef{\@glsref@#2}{}%
4854   \@for\@this@label:=\@glo@sortinglist\do{%

```

Has this entry already been added?

```

4855     \xifinlistcs{\@this@label}{\@glsref@#2}%
4856     {}%
4857     {%
4858       \listcsxadd{\@glsref@#2}{\@this@label}%
4859     }%
4860     \ifcsdef{\@glo@sortingchildren@\@this@label}%
4861     {%
4862       \@glo@addchildren{#2}{\@this@label}%
4863     }%
4864     {}%
4865   }%
4866 }

```

$\backslash@glo@addchildren$

$\backslash@glo@addchildren{\langle type \rangle}{\langle parent \rangle}$

```

4867 \newcommand*{\@glo@addchildren}[2]{%

```

Scope to allow nesting.

```

4868   \bgroup
4869   \letcs{\@glo@childlist}{\@glo@sortingchildren@#2}%
4870   \@for\@this@childlabel:=\@glo@childlist\do
4871   {%

```

Check this label hasn't already been added.

```

4872     \xifinlistcs{\@this@childlabel}{\@glsref@#1}%
4873     {}%
4874     {%
4875       \listcsxadd{\@glsref@#1}{\@this@childlabel}%
4876     }%

```

Does this child have children?

```

4877     \ifcsdef{\@glo@sortingchildren@\@this@childlabel}%
4878     {%
4879       \@glo@addchildren{#1}{\@this@childlabel}%
4880     }%

```



```

4881      {%
4882      }%
4883      }%
4884      \egroup
4885 }

```

@glo@do@sortentries

```

4886 \newcommand*{\@glo@do@sortentries}[1]{%
4887   \ifglshasparent{#1}%
4888   {%
      This entry has a parent, so add it to the child list
4889   \edef\@glo@parent{\csuse{glo@\glstoklabel{#1}@parent}}%
4890   \ifcsundef{\@glo@sortingchildren@\@glo@parent}%
4891   {%
4892     \csdef{\@glo@sortingchildren@\@glo@parent}{}%
4893   }%
4894   }%
4895   \expandafter\@glo@sortedinsert
4896   \csname @glo@sortingchildren@\@glo@parent\endcsname{#1}%

```

Has the parent been added?

```

4897   \xifinlistcs{\@glo@parent}{\glstref{\@glo@type}%
4898   {%

```

Yes, it has so do nothing.

```

4899   }%
4900   {%

```

No, it hasn't so add it now.

```

4901   \expandafter\@glo@do@sortentries\expandafter{\@glo@parent}%
4902   }%
4903   }%
4904   {%
4905   \@glo@sortedinsert{\@glo@sortinglist}{#1}%
4906   }%
4907 }

```

\@glo@sortedinsert \@glo@sortedinsert{<list>}{<entry label>}

Insert into list.

```

4908 \newcommand*{\@glo@sortedinsert}[2]{%
4909   \dtlinsertinto{#2}{#1}{\@glo@sortinghandler}%
4910 }%

```

The sort handlers need to be in the form required by datatool's \dtl@sortlist macro. These must set the count register \dtl@sortresult to either -1 (#1 less than #2), 0 (#1 = #2) or +1 (#1 greater than #2).

lo@sorthandler@word

```
4911 \newcommand*{\@glo@sorthandler@word}[2]{%
4912   \letcs\@gls@sort@A{glo\glsdetoklabel{#1}@sort}%
4913   \letcs\@gls@sort@B{glo\glsdetoklabel{#2}@sort}%
4914   \edef\glo@do@compare{%
4915     \noexpand\dtlwordindexcompare{\noexpand\dtl@sortresult}%
4916     {\expandonce\@gls@sort@B}%
4917     {\expandonce\@gls@sort@A}%
4918   }%
4919   \glo@do@compare
4920 }
```

@sorthandler@letter

```
4921 \newcommand*{\@glo@sorthandler@letter}[2]{%
4922   \letcs\@gls@sort@A{glo\glsdetoklabel{#1}@sort}%
4923   \letcs\@gls@sort@B{glo\glsdetoklabel{#2}@sort}%
4924   \edef\glo@do@compare{%
4925     \noexpand\dtlletterindexcompare{\noexpand\dtl@sortresult}%
4926     {\expandonce\@gls@sort@B}%
4927     {\expandonce\@gls@sort@A}%
4928   }%
4929   \glo@do@compare
4930 }
```

lo@sorthandler@case Case-sensitive sort.

```
4931 \newcommand*{\@glo@sorthandler@case}[2]{%
4932   \letcs\@gls@sort@A{glo\glsdetoklabel{#1}@sort}%
4933   \letcs\@gls@sort@B{glo\glsdetoklabel{#2}@sort}%
4934   \edef\glo@do@compare{%
4935     \noexpand\dtlcompare{\noexpand\dtl@sortresult}%
4936     {\expandonce\@gls@sort@B}%
4937     {\expandonce\@gls@sort@A}%
4938   }%
4939   \glo@do@compare
4940 }
```

@sorthandler@nocase Case-insensitive sort.

```
4941 \newcommand*{\@glo@sorthandler@nocase}[2]{%
4942   \letcs\@gls@sort@A{glo\glsdetoklabel{#1}@sort}%
4943   \letcs\@gls@sort@B{glo\glsdetoklabel{#2}@sort}%
4944   \edef\glo@do@compare{%
4945     \noexpand\dtlicompare{\noexpand\dtl@sortresult}%
4946     {\expandonce\@gls@sort@B}%
4947     {\expandonce\@gls@sort@A}%
4948   }%
4949   \glo@do@compare
4950 }
```

@glo@sortmacro@word Sort macro for ‘word’

```
4951 \newcommand*{\@glo@sortmacro@word}[1]{%
4952   \ifdefstring{\@glo@default@sorttype}{standard}%
4953   {%
4954     \@glo@sortentries{\@glo@sorthandler@word}{#1}%
4955   }%
4956   {%
4957     \PackageError{glossaries}{Conflicting sort options:^^J
4958       \string\usepackage[sort=\@glo@default@sorttype]{glossaries}^^J
4959       \string\printnoidxglossary[sort=word]}{}%
4960   }%
4961 }
```

@lo@sortmacro@letter Sort macro for ‘letter’

```
4962 \newcommand*{\@glo@sortmacro@letter}[1]{%
4963   \ifdefstring{\@glo@default@sorttype}{standard}%
4964   {%
4965     \@glo@sortentries{\@glo@sorthandler@letter}{#1}%
4966   }%
4967   {%
4968     \PackageError{glossaries}{Conflicting sort options:^^J
4969       \string\usepackage[sort=\@glo@default@sorttype]{glossaries}^^J
4970       \string\printnoidxglossary[sort=letter]}{}%
4971   }%
4972 }
```

@sortmacro@standard Sort macro for ‘standard’. (Use either ‘word’ or ‘letter’ order.)

```
4973 \newcommand*{\@glo@sortmacro@standard}[1]{%
4974   \ifdefstring{\@glo@default@sorttype}{standard}%
4975   {%
4976     \ifcsdef{\@glo@sorthandler@\glsorder}%
4977     {%
4978       \@glo@sortentries{\csuse{\@glo@sorthandler@\glsorder}}{#1}%
4979     }%
4980     {%
4981       \PackageError{glossaries}{Unknown sort handler ‘\glsorder’}{}%
4982     }%
4983   }%
4984   {%
4985     \PackageError{glossaries}{Conflicting sort options:^^J
4986       \string\usepackage[sort=\@glo@default@sorttype]{glossaries}^^J
4987       \string\printnoidxglossary[sort=standard]}{}%
4988   }%
4989 }
```

@glo@sortmacro@case Sort macro for ‘case’

```
4990 \newcommand*{\@glo@sortmacro@case}[1]{%
4991   \ifdefstring{\@glo@default@sorttype}{standard}%
4992   {%
```

```

4993 \glo@sortentries{\glo@sorthandler@case}{#1}%
4994 }%
4995 {%
4996 \PackageError{glossaries}{Conflicting sort options:^^J
4997 \string\usepackage[sort=\glo@default@sorttype]{glossaries}^^J
4998 \string\printnoidxglossary[sort=case]}{%
4999 }%
5000 }

\lo@sortmacro@nocase Sort macro for 'nocase'

5001 \newcommand*\glo@sortmacro@nocase[1]{%
5002 \ifdefstring{\glo@default@sorttype}{standard}%
5003 {%
5004 \glo@sortentries{\glo@sorthandler@nocase}{#1}%
5005 }%
5006 {%
5007 \PackageError{glossaries}{Conflicting sort options:^^J
5008 \string\usepackage[sort=\glo@default@sorttype]{glossaries}^^J
5009 \string\printnoidxglossary[sort=nocase]}{%
5010 }%
5011 }

\glo@sortmacro@def Sort macro for 'def'. The order of definition is given in \glo@list@<type>.

5012 \newcommand*\glo@sortmacro@def[1]{%
5013 \def\glo@sortinglist{}%
5014 \forlslentries[#1]{\gls@thislabel}%
5015 {%
5016 \xifinlistcs{\gls@thislabel}{\glsref@#1}%
5017 {%
5018 \listcsadd{\glo@sortinglist}{\gls@thislabel}%
5019 }%
5020 {%
5021 }%
5022 }%
5023 \cslet{\glsref@#1}{\glo@sortinglist}%
5024 }

\lo@sortmacro@def@do This won't include parent entries that haven't been referenced.

5025 \newcommand*\glo@sortmacro@def@do[1]{%
5026 \ifinlistcs{#1}{\glsref@\glo@type}%
5027 {}%
5028 {%
5029 \listcsadd{\glsref@\glo@type}{#1}%
5030 }%
5031 \ifcsdef{\glo@sortingchildren@#1}%
5032 {%
5033 \glo@addchildren{\glo@type}{#1}%

```

```

5034 }%
5035 {}%
5036 }

\@glo@sortmacro@use Sort macro for 'use'. (No sorting is required, as the entries are already in order
of use, so do nothing.)
5037 \newcommand*{\@glo@sortmacro@use}[1]{%

\print@noidx@glossary Glossary handler for \printnoidxglossary which doesn't use an indexing ap-
plication. Since \printnoidxglossary may occur at the start of the docu-
ment, we can't just check if an entry has been used. Instead, the first pass needs
to write information to the aux file every time an entry is referenced. This needs
to be read in on the second run and stored in a list corresponding to the appro-
priate glossary.
5038 \newcommand*{\@print@noidx@glossary}{%
5039 \ifcsdef{@glsref@ \@glo@type}%
5040 {%
Sort the entries:
5041 \ifcsdef{@glo@sortmacro@ \@glo@sorttype}%
5042 {%
5043 \csuse{@glo@sortmacro@ \@glo@sorttype}{\@glo@type}%
5044 }%
5045 {%
5046 \PackageError{glossaries}{Unknown sort handler '@@glo@sorttype'}{ }%
5047 }%
Do the glossary heading and preamble
5048 \glossarysection[\glossarytoctitle]{\glossarytitle}%
5049 \glossarypreamble
5050 \begin{theglossary}%
5051 \glossaryheader
5052 \glsresetentrylist
5053 \def\@gls@currentlettergroup{}%
Iterate through the entries.
5054 \forlistcsloop{\@gls@noidx@do}{\@glsref@ \@glo@type}%
Finally end the glossary and do the postamble:
5055 \end{theglossary}%
5056 \glossarypostamble
5057 }%
5058 {%
5059 \@gls@noref@warn{\@glo@type}%
5060 }%
5061 }

\glo@grabfirst
5062 \def\glo@grabfirst#1#2\@nil{%
5063 \def\@gls@firsttok{#1}%

```

```

5064 \ifdefempty\@gls@firsttok
5065 {%
5066   \def\@glo@thislettergrp{0}%
5067 }%
5068 {%

  Sanitize it:
5069   \@onelevel@sanitize\@gls@firsttok

  Fetch the first letter:
5070   \expandafter\@glo@grabfirst\@gls@firsttok{}\}\@nil
5071 }%
5072 }

```

\@glo@grabfirst

```

5073 \def\@glo@grabfirst#1#2\@nil{%
5074   \ifdefempty\@glo@thislettergrp
5075   {%
5076     \def\@glo@thislettergrp{glssymbols}%
5077   }%
5078   {%
5079     \count@=\uccode'#1\relax
5080     \ifnum\count@=0\relax
5081       \def\@glo@thislettergrp{glssymbols}%
5082     \else
5083       \ifdefstring\@glo@sorttype{case}%
5084       {%
5085         \count@='#1\relax
5086       }%
5087       {%
5088       }%
5089       \edef\@glo@thislettergrp{\the\count@}%
5090     \fi
5091   }%
5092 }

```

\@gls@noidx@do Handler for list iteration used by \@print@noidx@glossary. The argument is the entry label. This only allows one sublevel.

```

5093 \newcommand{\@gls@noidx@do}[1]{%

  Get this entry's location list
5094   \global\letcs{\@gls@loclist}{glo@glstdetoklabel{#1}@loclist}%

  Does this entry have a parent?
5095   \ifglshasparent{#1}%
5096   {%

    Has a parent.
5097     \gls@level=\csuse{glo@glstdetoklabel{#1}@level}\relax
5098     \ifdefvoid{\@gls@loclist}
5099     {%

```

```

5100     \subglossentry{\gls@level}{#1}{}%
5101   }%
5102   {%
5103     \subglossentry{\gls@level}{#1}%
5104     {%
5105       \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
5106     }%
5107   }%
5108 }%
5109 {%

```

Doesn't have a parent Get this entry's sort key

```

5110   \letcs{\@gls@sort}{glo@glsdetoklabel{#1}@sort}%

```

Fetch the first letter:

```

5111   \expandafter\glo@grabfirst\@gls@sort{}{}\@nil
5112   \ifdefequal{\@glo@thislettergrp}{\@gls@currentlettergroup}%
5113   {}%
5114   {%

```

Do the group header:

```

5115     \ifdefempty{\@gls@currentlettergroup}{\@gls@groupskip}%
5116     \gls@groupheading{\@glo@thislettergrp}%
5117   }%
5118   \let\@gls@currentlettergroup\@glo@thislettergrp

```

Do this entry:

```

5119   \ifdefvoid{\@gls@loclist}
5120   {%
5121     \glossentry{#1}{}%
5122   }%
5123   {%
5124     \glossentry{#1}%
5125     {%
5126       \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
5127     }%
5128   }%
5129 }%
5130 }

```

\glsnoidxloclist `\glsnoidxloclist{<list cs>}`

Display location list.

```

5131 \newcommand*{\glsnoidxloclist}[1]{%
5132   \def\@gls@noidxloclist@sep{}%
5133   \def\@gls@noidxloclist@prev{}%
5134   \forlistloop{\glsnoidxloclisthandler}{#1}%
5135 }

```

noidxloclisthandler Handler for location list iterator.

```
5136 \newcommand*{\glsnoidxloclisthandler}[1]{%
5137   \ifdefstring{\@gls@noidxloclist@prev}{#1}%
5138   {%
      Same as previous location so skip.
5139   }%
5140   {%
5141     \@gls@noidxloclist@sep
5142     #1%
5143     \def\@gls@noidxloclist@sep{\delimN}%
5144     \def\@gls@noidxloclist@prev{#1}%
5145   }%
5146 }
```

splayloclisthandler Handler for location list iterator when used with \glsdisplaynumberlist.

```
5147 \newcommand*{\glsnoidxdisplayloclisthandler}[1]{%
5148   \ifdefstring{\@gls@noidxloclist@prev}{#1}%
5149   {%
      Same as previous location so skip.
5150   }%
5151   {%
5152     \@gls@noidxloclist@sep
5153     \@gls@noidxloclist@prev
5154     \def\@gls@noidxloclist@prev{#1}%
5155   }%
5156 }
```

\glsnoidxdisplayloc `\glsnoidxdisplayloc{<prefix>}{<counter>}{<format>}{<location>}`

Display a location in the location list.

```
5157 \newcommand*\glsnoidxdisplayloc[4]{%
5158   \setentrycounter{#1}{#2}%
5159   \csuse{#3}{#4}%
5160 }
```

\@gls@reference `\@gls@reference{<type>}{<label>}{<loc>}`

Identifies that a reference has been used (for use in the aux file). All entries must be defined in the preamble.

```
5161 \newcommand*\@gls@reference[3]{%
      Add to label list
5162   \glsdoifexistsorwarn{#2}%
5163   {%
5164     \ifcsundef{\@glsref@#1}{\csgdef{\@glsref@#1}{}}{}%

```



```

5165 \ifinlistcs{#2}{@glsref@#1}%
5166 {}%
5167 {\listcsgadd{@glsref@#1}{#2}}%
  Add to location list
5168 \ifcsundef{glo@glstdetoklabel{#2}@loclist}%
5169 {\csgdef{glo@glstdetoklabel{#2}@loclist}{}}%
5170 {}%
5171 \listcsgadd{glo@glstdetoklabel{#2}@loclist}{#3}%
5172 }%
5173 }

```

The keys that can be used in the optional argument to `\printglossary` or `\printnoidxglossary` are as follows: The `type` key sets the glossary type.

```

5174 \define@key{printgloss}{type}{\def@glo@type{#1}}

```

The `title` key sets the title used in the glossary section header. This overrides the title used in `\newglossary`.

```

5175 \define@key{printgloss}{title}{%
5176 \def@glossarytitle{#1}%
5177 \let@gls@dotoc@title\relax
5178 }

```

The `toctitle` sets the text used for the relevant entry in the table of contents.

```

5179 \define@key{printgloss}{toctitle}{%
5180 \def@glossarytoctitle{#1}%
5181 \let@gls@dotoc@title\relax
5182 }

```

The `style` key sets the glossary style (but only for the given glossary).

```

5183 \define@key{printgloss}{style}{%
5184 \ifcsundef{@glsstyle@#1}%
5185 {%
5186 \PackageError{glossaries}%
5187 {Glossary style ‘#1’ undefined}{}%
5188 }%
5189 {%
5190 \def@glossarystyle{\setglossentrycompatibility
5191 \csname @glsstyle@#1\endcsname}%
5192 }%
5193 }

```

The `numberedsection` key determines if this glossary should be in a numbered section.

```

5194 \define@choicekey{printgloss}{numberedsection}[\val\nr]{%
5195 false,nolabel,autolabel,nameref}[nolabel]{%
5196 \ifcase\nr\relax
5197 \renewcommand*{\@@glossarysecstar}{*}%
5198 \renewcommand*{\@@glossaryseclabel}{}%
5199 \or
5200 \renewcommand*{\@@glossarysecstar}{}%

```

```

5201 \renewcommand*{\@@glossaryseclabel}{}%
5202 \or
5203 \renewcommand*{\@@glossarysecstar}{}%
5204 \renewcommand*{\@@glossaryseclabel}{\label{\glsautoprefix\@glo@type}}%
5205 \or
5206 \renewcommand*{\@@glossarysecstar}{*}%
5207 \renewcommand*{\@@glossaryseclabel}{%
5208 \protected@edef\@currentlabelname{\glossarytoctitle}%
5209 \label{\glsautoprefix\@glo@type}}%
5210 \fi
5211 }

```

The `nogroupskip` key determines whether or not there should be a vertical gap between glossary groups.

```

5212 \define@choicekey{printgloss}{nogroupskip}{true,false}[true]{%
5213 \csuse{glsnogroupskip#1}%
5214 }

```

The `nopostdot` key has the same effect as the package option of the same name.

```

5215 \define@choicekey{printgloss}{nopostdot}{true,false}[true]{%
5216 \csuse{glsnopostdot#1}%
5217 }

```

The `entrycounter` key is the same as the package option but localised to the current glossary.

```

5218 \define@choicekey{printgloss}{entrycounter}{true,false}[true]{%
5219 \csuse{glsentrycounter#1}%
5220 \ifglsentrycounter
5221 \ifx\@gls@counterwithin\@empty
5222 \newcounter{glossaryentry}%
5223 \else
5224 \newcounter{glossaryentry}[\@gls@counterwithin]%
5225 \fi
5226 \def\theHglossaryentry{\currentglossary.\theglossaryentry}%
5227 \renewcommand*{\glsresetentrycounter}{%
5228 \setcounter{glossaryentry}{0}%
5229 }%
5230 \renewcommand*{\glsstepentry}[1]{%
5231 \refstepcounter{glossaryentry}%
5232 \label{glsentry-\glsdetoklabel{##1}}%
5233 }%
5234 \renewcommand*{\glsentrycounterlabel}{\theglossaryentry.\space}%
5235 \renewcommand*{\glsentryitem}[1]{%
5236 \glsstepentry{##1}\glsentrycounterlabel
5237 }%
5238 \else
5239 \renewcommand*{\glsresetentrycounter}{}%
5240 \renewcommand*{\glsstepentry}[1]{%
5241 \renewcommand*{\glsentrycounterlabel}{}%

```

```

5242 \renewcommand*{\glsubentryitem}[1]{\glresetsubentrycounter}
5243 \fi
5244 }

```

The subentrycounter key is the same as the package option but localised to the current glossary. Note that this doesn't affect the master/slave counter attributes, which occurs if subentrycounter and entrycounter package options are set to true.

```

5245 \define@choicekey{printgloss}{subentrycounter}{true,false}[true]{%
5246 \csuse{glssubentrycounter#1}%
5247 \ifglssubentrycounter
5248 \ifundef\c@glossarysubentry
5249 {%
5250 \ifglsubentrycounter
5251 \newcounter{glossarysubentry}[glossaryentry]%
5252 \else
5253 \newcounter{glossarysubentry}
5254 \fi
5255 }{}%
5256 \renewcommand*{\glssubentry}[1]{%
5257 \edef\currentglssubentry{\glsetoklabel{##1}}%
5258 \refstepcounter{glossarysubentry}%
5259 \label{glsubentry-\currentglssubentry}%
5260 }%
5261 \renewcommand*{\glresetsubentrycounter}{%
5262 \setcounter{glossarysubentry}{0}%
5263 }%
5264 \renewcommand*{\glssubentryitem}[1]{%
5265 \glssubentry{##1}\glssubentrycounterlabel
5266 }%
5267 \renewcommand*{\glssubentrycounterlabel}{\theglossarysubentry}\space}%
5268 \def\theHglossarysubentry{\currentglssubentry.\theglossarysubentry}
5269 \else
5270 \renewcommand*{\glssubentryitem}[1]{}%
5271 \renewcommand*{\glssubentry}[1]{}%
5272 \renewcommand*{\glresetsubentrycounter}{}%
5273 \renewcommand*{\glssubentrycounterlabel}{}%
5274 \fi
5275 }

```

The nonnumberlist key determines if this glossary should have a number list.

```

5276 \define@boolkey{printgloss}[gls]{nonnumberlist}[true]{%
5277 \ifglsnonnumberlist
5278 \def\glossaryentrynumbers##1{}%
5279 \else
5280 \def\glossaryentrynumbers##1{##1}%
5281 \fi}

```

The sort key sets the glossary sort handler (\printnoidxglossary only).

```

5282 \define@key{printgloss}{sort}{\@glo@assign@sortkey{#1}}

```

`\@no@assign@sortkey` Issue error if used with `\printglossary`

```

5283 \newcommand*{\@glo@no@assign@sortkey}[1]{%
5284   \PackageError{glossaries}{‘sort’ key not permitted with
5285   \string\printglossary}%
5286   {The ‘sort’ key may only be used with \string\printnoidxglossary}%
5287 }

```

`\@glo@assign@sortkey` For use with `\printnoidxglossary`

```

5288 \newcommand*{\@glo@assign@sortkey}[1]{%
5289   \def\@glo@sorttype{#1}%
5290 }

```

`\@glsnonextpages` Suppresses the next number list only. Global assignments required as it may not occur in the same level of grouping as the next numberlist. (For example, if `\glsnonextpages` is place in the entry’s description and 3 column tabular style glossary is used.) `\org@glossaryentrynumbers` needs to be set at the start of each glossary, in the event that `\glossaryentrynumber` is redefined.

```

5291 \newcommand*{\@glsnonextpages}{%
5292   \gdef\glossaryentrynumbers##1{%
5293     \glsresetentrylist
5294   }%
5295 }

```

`\@glsnextpages` Activate the next number list only. Global assignments required as it may not occur in the same level of grouping as the next numberlist. (For example, if `\glsnextpages` is place in the entry’s description and 3 column tabular style glossary is used.) `\org@glossaryentrynumbers` needs to be set at the start of each glossary, in the event that `\glossaryentrynumber` is redefined.

```

5296 \newcommand*{\@glsnextpages}{%
5297   \gdef\glossaryentrynumbers##1{%
5298     ##1\glsresetentrylist}}

```

`\glsresetentrylist` Resets `\glossaryentrynumbers`

```

5299 \newcommand*{\glsresetentrylist}{%
5300   \global\let\glossaryentrynumbers\org@glossaryentrynumbers}

```

`\glsnonextpages` Outside of `\printglossary` this does nothing.

```

5301 \newcommand*{\glsnonextpages}{}

```

`\glsnextpages` Outside of `\printglossary` this does nothing.

```

5302 \newcommand*{\glsnextpages}{}

```

`glossaryentry` If the `entrycounter` package option has been used, define a counter to number each level 0 entry.

```

5303 \ifglentrycounter
5304   \ifx\@gls@counterwithin\@empty
5305     \newcounter{glossaryentry}

```

```

5306 \else
5307   \newcounter{glossaryentry}[\@gls@counterwithin]
5308 \fi
5309 \def\theHglossaryentry{\currentglossary.\theglossaryentry}
5310 \fi

```

glossarysubentry If the subentrycounter package option has been used, define a counter to number each level 1 entry.

```

5311 \ifglssubentrycounter
5312   \ifglentrycounter
5313     \newcounter{glossarysubentry}[glossaryentry]
5314   \else
5315     \newcounter{glossarysubentry}
5316   \fi
5317 \def\theHglossarysubentry{\currentglssubentry.\theglossarysubentry}
5318 \fi

```

resetsubentrycounter Resets the glossarysubentry counter.

```

5319 \ifglssubentrycounter
5320   \newcommand*{\glsresetsubentrycounter}{%
5321     \setcounter{glossarysubentry}{0}%
5322   }
5323 \else
5324   \newcommand*{\glsresetsubentrycounter}{}
5325 \fi

```

resetentrycounter Resets the glossaryentry counter.

```

5326 \ifglentrycounter
5327   \newcommand*{\glsresetentrycounter}{%
5328     \setcounter{glossaryentry}{0}%
5329   }
5330 \else
5331   \newcommand*{\glsresetentrycounter}{}
5332 \fi

```

\glsstepentry Advance the glossaryentry counter if in use. The argument is the label associated with the entry.

```

5333 \ifglentrycounter
5334   \newcommand*{\glsstepentry}[1]{%
5335     \refstepcounter{glossaryentry}%
5336     \label{glentry-\glsdetoklabel{#1}}%
5337   }
5338 \else
5339   \newcommand*{\glsstepentry}[1]{}
5340 \fi

```

\glsstepsubentry Advance the glossarysubentry counter if in use. The argument is the label associated with the subentry.

```

5341 \ifglssubentrycounter
5342   \newcommand*{\glssubentry}[1]{%
5343     \edef\currentglssubentry{\glsdetoklabel{#1}}%
5344     \refstepcounter{glossarysubentry}%
5345     \label{glssentry-\currentglssubentry}%
5346   }
5347 \else
5348   \newcommand*{\glssubentry}[1]{}
5349 \fi

```

`\glssrefentry` Reference the entry or sub-entry counter if in use, otherwise just do `\gls`.

```

5350 \ifglssentrycounter
5351   \newcommand*{\glssrefentry}[1]{\ref{glssentry-\glsdetoklabel{#1}}}
5352 \else
5353   \ifglssubentrycounter
5354     \newcommand*{\glssrefentry}[1]{\ref{glssentry-\glsdetoklabel{#1}}}
5355   \else
5356     \newcommand*{\glssrefentry}[1]{\gls{#1}}
5357   \fi
5358 \fi

```

`\glssentrycounterlabel` Defines how to display the glossaryentry counter.

```

5359 \ifglssentrycounter
5360   \newcommand*{\glssentrycounterlabel}{\theglossaryentry.\space}
5361 \else
5362   \newcommand*{\glssentrycounterlabel}{}
5363 \fi

```

`\glssubentrycounterlabel` Defines how to display the glossarysubentry counter.

```

5364 \ifglssubentrycounter
5365   \newcommand*{\glssubentrycounterlabel}{\theglossarysubentry)\space}
5366 \else
5367   \newcommand*{\glssubentrycounterlabel}{}
5368 \fi

```

`\glssentryitem` Step and display glossaryentry counter, if appropriate.

```

5369 \ifglssentrycounter
5370   \newcommand*{\glssentryitem}[1]{%
5371     \glssstepentry{#1}\glssentrycounterlabel
5372   }
5373 \else
5374   \newcommand*{\glssentryitem}[1]{\glssresetsubentrycounter}
5375 \fi

```

`\glsssubentryitem` Step and display glossarysubentry counter, if appropriate.

```

5376 \ifglsssubentrycounter
5377   \newcommand*{\glsssubentryitem}[1]{%
5378     \glssstepsubentry{#1}\glsssubentrycounterlabel
5379   }

```

```

5380 \else
5381   \newcommand*{\glssubentryitem}[1]{%
5382 \fi

```

theglossary If the theglossary environment has already been defined, a warning will be issued. This environment should be redefined by glossary styles.

```

5383 \ifcsundef{theglossary}%
5384 {%
5385   \newenvironment{theglossary}{}{}%
5386 }%
5387 {%
5388   \@gls@warnontheglossdefined
5389   \renewenvironment{theglossary}{}{}%
5390 }

```

The glossary header is given by `\glossaryheader`. This forms part of the glossary style, and must indicate what should appear immediately after the start of the theglossary environment. (For example, if the glossary uses a tabular-like environment, it may be used to set the header row.) Note that if you don't want a header row, the glossary style must redefine `\glossaryheader` to do nothing.

`\glossaryheader`

```

5391 \newcommand*{\glossaryheader}{}

```

`\glstarget` `\glstarget{<label>}{<name>}`

Provide user interface to `\@glstarget` to make it easier to modify the glossary style in the document.

```

5392 \newcommand*{\glstarget}[2]{\@glstarget{\glo@linkprefix#1}{#2}}

```

As from version 3.08, glossary information is now written to the external files using `\glossentry` and `\subglossentry` instead of `\glossaryentryfield` and `\glossarysubentryfield`. The default definition provides backward compatibility for glossary styles that use the old forms.

`compatibleglossentry`

`\glossentry{<label>}{<page-list>}`

```

5393 \providecommand*{\compatibleglossentry}[2]{%
5394   \toks@{#2}%
5395   \protected@edef\@do@glossentry{\noexpand\glossaryentryfield{#1}%
5396     {\noexpand\glsnamefont
5397       {\expandafter\expandonce\csname glo@#1@name\endcsname}}}%
5398     {\expandafter\expandonce\csname glo@#1@desc\endcsname}}%
5399     {\expandafter\expandonce\csname glo@#1@symbol\endcsname}}%
5400   {\the\toks@}%
5401 }%

```

```

5402 \@do@glossentry
5403 }

```

\glossentryname

```

5404 \newcommand*{\glossentryname}[1]{%
5405   \glsdoifexistsorwarn{#1}%
5406   {%
5407     \letcs{\glo@name}{\glo@\glsdetoklabel{#1}@name}%
5408     \expandafter\glsnamefont\expandafter{\glo@name}%
5409   }%
5410 }

```

\Glossentryname

```

5411 \newcommand*{\Glossentryname}[1]{%
5412   \glsdoifexistsorwarn{#1}%
5413   {%
5414     \glsnamefont{\Glsentryname{#1}}%
5415   }%
5416 }

```

\glossentrydesc

```

5417 \newcommand*{\glossentrydesc}[1]{%
5418   \glsdoifexistsorwarn{#1}%
5419   {%
5420     \glentrydesc{#1}%
5421   }%
5422 }

```

\Glossentrydesc

```

5423 \newcommand*{\Glossentrydesc}[1]{%
5424   \glsdoifexistsorwarn{#1}%
5425   {%
5426     \Glsentrydesc{#1}%
5427   }%
5428 }

```

\glossentrysymbol

```

5429 \newcommand*{\glossentrysymbol}[1]{%
5430   \glsdoifexistsorwarn{#1}%
5431   {%
5432     \glentrysymbol{#1}%
5433   }%
5434 }

```

\Glossentrysymbol

```

5435 \newcommand*{\Glossentrysymbol}[1]{%
5436   \glsdoifexistsorwarn{#1}%
5437   {%

```



```

5438     \Glsentrysymbol{#1}%
5439 }%
5440 }

```

compatiblesubglossentry `\subglossentry{<level>}{<label>}{<page-list>}`

```

5441 \providecommand*\compatiblesubglossentry}[3]{%
5442   \toks@{#3}%
5443   \protected@edef\@do@subglossentry{\noexpand\glossarysubentryfield{\number#1}%
5444     {#2}%
5445     {\noexpand\glsnamefont
5446       {\expandafter\expandonce\csname glo@#2@name\endcsname}}%
5447     {\expandafter\expandonce\csname glo@#2@desc\endcsname}%
5448     {\expandafter\expandonce\csname glo@#2@symbol\endcsname}%
5449     {\the\toks@}}%
5450 }%
5451 \@do@subglossentry
5452 }

```

sentrycompatibility

```

5453 \newcommand*\setglossentrycompatibility{%
5454   \let\glossentry\compatibleglossentry
5455   \let\subglossentry\compatiblesubglossentry
5456 }
5457 \setglossentrycompatibility

```

\glossaryentryfield

`\glossaryentryfield{<label>}{<name>}{<description>}{<symbol>}{<page-list>}`

This command formerly governed how each entry row should be formatted in the glossary. Now deprecated.

```

5458 \newcommand{\glossaryentryfield}[5]{%
5459   \GlossariesWarning
5460   {Deprecated use of \string\glossaryentryfield.^^J
5461     I recommend you change to \string\glossentry.^^J
5462     If you've just upgraded, try removing your gls auxiliary
5463     files^^J and recompile}%
5464   \noindent\textbf{\glstarget{#1}{#2}} #4 #3. #5\par}

```

lossarysubentryfield

`\glossarysubentryfield{<level>}{<label>}{<name>}{<description>}{<symbol>}{<page-list>}`

This command governs how each subentry should be formatted in the glossary. Glossary styles need to redefine this command. Most of the predefined styles

ignore $\langle symbol \rangle$. The first argument is a number indicating the level. (The level should be greater than or equal to 1.)

```
5465 \newcommand*{\glossarysubentryfield}[6]{%
5466   \GlossariesWarning
5467   {Deprecated use of \string\glossarysubentryfield.^^J
5468    I recommend you change to \string\subglossentry.^^J
5469    If you've just upgraded, try removing your gls auxiliary
5470    files^^J and recompile}%
5471   \glstarget{#2}{\strut}#4. #6\par}
```

Within each glossary, the entries form distinct groups which are determined by the first character of the sort key. When using `makeindex`, there will be a maximum of 28 groups: symbols, numbers, and the 26 alphabetical groups A, ..., Z. If you use `xindy` the groups will depend on whatever alphabet is used. This is determined by the language or custom alphabets can be created in the `xindy` style file. The command `\glsgroupskip` specifies what to do between glossary groups. Glossary styles must redefine this command. (Note that `\glsgroupskip` only occurs between groups, not at the start or end of the glossary.)

`\glsgroupskip`

```
5472 \newcommand*{\glsgroupskip}{}
```

Each of the 28 glossary groups described above is preceded by a group heading. This is formatted by the command `\glsgroupheading` which takes one argument which is the *label* assigned to that group (not the title). The corresponding labels are: `glssymbols`, `glsnumbers`, A, ..., Z. Glossary styles must redefine this command. (In between groups, `\glsgroupheading` comes immediately after `\glsgroupskip`.)

`\glsgroupheading`

```
5473 \newcommand*{\glsgroupheading}[1]{}
```

It is possible to “trick” `makeindex` into treating entries as though they belong to the same group, even if the terms don’t start with the same letter, by modifying the sort key. For example, all entries belonging to one group could be defined so that the sort key starts with an a, while entries belonging to another group could be defined so that the sort key starts with a b, and so on. If you want each group to have a heading, you would then need to modify the translation control sequences `\glsgetgrouptitle` and `\glsgetgrouplabel` so that the label is translated into the required title (and vice-versa).

`\glsgetgrouptitle{<label>}`

This command produces the title for the glossary group whose label is given by $\langle label \rangle$. By default, the group labelled `glssymbols` produces `\glssymbolsgroupname`, the group labelled `glsnumbers` produces `\glsnumbersgroupname` and all the

other groups simply produce their label. As mentioned above, the group labels are: `glsymbols`, `glsnumbers`, A, ..., Z. If you want to redefine the group titles, you will need to redefine this command. Languages other than English may produce labels that are non-expandable, so we need to check for that otherwise it will create a “missing `\endcsname` inserted” error.

`\glsgetgrouptitle`

```
5474 \newcommand*{\glsgetgrouptitle}[1]{%
5475   \@gls@getgrouptitle{#1}{\@gls@grptitle}%
5476   \@gls@grptitle
5477 }
```

`\@gls@getgrouptitle` Gets the group title specified by the label (first argument) and stores in the second argument, which must be a control sequence.

```
5478 \newcommand*{\@gls@getgrouptitle}[2]{%
```

Even if the argument appears to be a single letter, it won't be considered a single letter by `\dtl@ifsingle` if it's an active character.

```
5479 \dtl@ifsingle{#1}%
5480 {%
5481   \ifcsundef{#1groupname}{\def#2{#1}}{\letcs#2{#1groupname}}%
5482 }%
5483 {%
5484   \ifboolexpr{test{\ifstrequal{#1}{glsymbols}}
5485               or test{\ifstrequal{#1}{glsnumbers}}}%
5486   {%
5487     \ifcsundef{#1groupname}{\def#2{#1}}{\letcs#2{#1groupname}}%
5488   }%
5489   {%
5490     \def#2{#1}%
5491   }%
5492 }%
5493 }
```

`@getothergrouptitle` Version for the no-indexing app option:

```
5494 \newcommand*{\@gls@noidx@getgrouptitle}[2]{%
5495   \DTLifint{#1}%
5496   {\edef#2{\char#1\relax}}%
5497   {%
5498     \ifcsundef{#1groupname}{\def#2{#1}}{\letcs#2{#1groupname}}%
5499   }%
5500 }
```

`\glsgetgrouplabel{<title>}`

This command does the reverse to the previous command. The argument is the group title, and it produces the group label. Note that if you redefine `\glsgetgrouptitle`, you will also need to redefine `\glsgetgrouplabel`.

`\glsgetgrouplabel`

```
5501 \newcommand*{\glsgetgrouplabel}[1]{%
5502 \ifthenelse{\equal{#1}{\glssymbolsgroupname}}{\glssymbols}{%
5503 \ifthenelse{\equal{#1}{\glsnumbersgroupname}}{\glsnumbers}{#1}}}
```

The command `\setentrycounter` sets the entry's associated counter (required by `\glshypernumber` etc.) `\glslink` and `\glsadd` encode the `\glossary` argument so that the relevant counter is set prior to the formatting command.

`\setentrycounter`

```
5504 \newcommand*{\setentrycounter}[2][1]{%
5505   \def\@glo@counterprefix{#1}%
5506   \ifx\@glo@counterprefix\@empty
5507     \def\@glo@counterprefix{.}%
5508   \else
5509     \def\@glo@counterprefix{.#1.}%
5510   \fi
5511   \def\glsentrycounter{#2}%
5512 }
```

The current glossary style can be set using `\setglossarystyle{<style>}`.

`\setglossarystyle`

```
5513 \newcommand*{\setglossarystyle}[1]{%
5514   \ifcsundef{@glsstyle@#1}%
5515   {%
5516     \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
5517   }%
5518   {%
5519     \csname @glsstyle@#1\endcsname
5520   }%
5521 }
```

`\glossarystyle`

```
5522 \newcommand*{\glossarystyle}[1]{%
5523   \ifcsundef{@glsstyle@#1}%
5524   {%
5525     \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
5526   }%
5527   {%
5528     \GlossariesWarning
5529     {Deprecated command \string\glossarystyle.^^J
5530     I recommend you switch to \string\setglossarystyle\space unless
5531     you want to maintain backward compatibility}%
5532     \setglossentrycompatibility
5533     \csname @glsstyle@#1\endcsname

5534     \ifcsdef{@glscompstyle@#1}%
5535     {\setglossentrycompatibility\csuse{@glscompstyle@#1}}%
5536     {}%
```

```
5537 }%
5538 }
```

`\newglossarystyle` New glossary styles can be defined using:

```
\newglossarystyle{<name>}{<definition>}
```

The *<definition>* argument should redefine `\theglossary`, `\glossaryheader`, `\glsgroupheading`, `\glossaryentryfield` and `\glsgroupskip` (see [subsection 1.18](#) for the definitions of predefined styles). Glossary styles should not redefine `\glossarypreamble` and `\glossarypostamble`, as the user should be able to switch between styles without affecting the pre- and postambles.

```
5539 \newcommand{\newglossarystyle}[2]{%
5540   \ifcsundef{@glsstyle@#1}%
5541   {%
5542     \expandafter\def\csname @glsstyle@#1\endcsname{#2}%
5543   }%
5544   {%
5545     \PackageError{glossaries}{Glossary style ‘#1’ is already defined}{}%
5546   }%
5547 }
```

`\renewglossarystyle` Code for this macro supplied by Marco Daniel.

```
5548 \newcommand{\renewglossarystyle}[2]{%
5549   \ifcsundef{@glsstyle@#1}%
5550   {%
5551     \PackageError{glossaries}{Glossary style ‘#1’ isn’t already defined}{}%
5552   }%
5553   {%
5554     \csdef{@glsstyle@#1}{#2}%
5555   }%
5556 }
```

Glossary entries are encoded so that the second argument to `\glossaryentryfield` is always specified as `\glsnamefont{<name>}`. This allows the user to change the font used to display the name term without having to redefine `\glossaryentryfield`. The default uses the surrounding font, so in the list type styles (which place the name in the optional argument to `\item`) the name will appear in bold.

`\glsnamefont`

```
5557 \newcommand*{\glsnamefont}[1]{#1}
```

Each glossary entry has an associated number list (usually page numbers) that indicate where in the document the entry has been used. The format for these number lists can be changed using the format key in commands like `\glslink`. The default format is given by `\glshypernumber`. This takes a single argument which may be a single number, a number range or a number list.

The number ranges are delimited with `\delimR`, the number lists are delimited with `\delimN`.

If the document doesn't have hyperlinks, the numbers can be displayed just as they are, but if the document supports hyperlinks, the numbers should link to the relevant location. This means extracting the individual numbers from the list or ranges. The package does this with the `\hyperpage` command, but this is encoded for comma and dash delimiters and only for the page counter, but this code needs to be more general. So I have adapted the code used in the package.

`\glshypernumber`

```
5558 \ifcsundef{hyperlink}%
5559 {%
5560   \def\glshypernumber#1{#1}%
5561 }%
5562 {%
5563   \def\glshypernumber#1{\@glshypernumber#1\nohyperpage{}}\@nil}
5564 }
```

`\@glshypernumber` This code was provided by Heiko Oberdiek to allow material to be attached to the location.

```
5565 \def\@glshypernumber#1\nohyperpage#2#3\@nil{%
5566   \ifx\#1\%
5567     \else
5568       \@delimR#1\delimR\delimR\%
5569     \fi
5570     \ifx\#2\%
5571       \else
5572         #2%
5573       \fi
5574       \ifx\#3\%
5575         \else
5576           \@glshypernumber#3\@nil
5577         \fi
5578 }
```

`\@delimR` displays a range of numbers for the counter whose name is given by `\@gls@counter` (which must be set prior to using `\glshypernumber`).

`\@delimN`

```
5579 \def\@delimR#1\delimR #2\delimR #3\%
5580 \ifx\#2\%
5581   \@delimN{#1}%
5582 \else
5583   \@gls@numberlink{#1}\delimR\@gls@numberlink{#2}%
5584 \fi}
```

`\@delimN` displays a list of individual numbers, instead of a range:

\@delimN

```
5585 \def\@delimN#1{\@delimN#1\delimN \delimN\\}
5586 \def\@delimN#1\delimN #2\delimN#3\\{\%
5587 \ifx\\#3\\%
5588   \@gls@numberlink{#1}%
5589 \else
5590   \@gls@numberlink{#1}\delimN\@gls@numberlink{#2}%
5591 \fi
5592 }
```

The following code is modified from hyperref's \HyInd@pagelink where the name of the counter being used is given by \@gls@counter.

```
5593 \def\@gls@numberlink#1{%
5594 \begingroup
5595   \toks@={}%
5596   \@gls@removespaces#1 \@nil
5597 \endgroup}

5598 \def\@gls@removespaces#1 #2\@nil{%
5599   \toks@=\expandafter{\the\toks@#1}%
5600   \ifx\\#2\\%
5601     \edef\x{\the\toks@}%
5602     \ifx\x\empty
5603       \else

5604         \hyperlink{\glstrycounter\@glo@counterprefix\the\toks@}%
5605           {\the\toks@}%
5606     \fi
5607   \else
5608     \@gls@ReturnAfterFi{%
5609       \@gls@removespaces#2\@nil
5610     }%
5611   \fi
5612 }
5613 \long\def\@gls@ReturnAfterFi#1\fi{\fi#1}
```

The following commands will switch to the appropriate font, and create a hyperlink, if hyperlinks are supported. If hyperlinks are not supported, they will just display their argument in the appropriate font.

\hyperrm

```
5614 \newcommand*\hyperrm[1]{\textrm{\glshypernumber{#1}}}
```

\hypersf

```
5615 \newcommand*\hypersf[1]{\textsf{\glshypernumber{#1}}}
```

\hypertt

```
5616 \newcommand*\hypertt[1]{\texttt{\glshypernumber{#1}}}
```

```

\hyperbf
5617 \newcommand*{\hyperbf}[1]{\textbf{\glshypernumber{#1}}}

\hypermd
5618 \newcommand*{\hypermd}[1]{\textmd{\glshypernumber{#1}}}

\hyperit
5619 \newcommand*{\hyperit}[1]{\textit{\glshypernumber{#1}}}

\hypersl
5620 \newcommand*{\hypersl}[1]{\textsl{\glshypernumber{#1}}}

\hyperup
5621 \newcommand*{\hyperup}[1]{\textup{\glshypernumber{#1}}}

\hypersc
5622 \newcommand*{\hypersc}[1]{\textsc{\glshypernumber{#1}}}

\hyperemph
5623 \newcommand*{\hyperemph}[1]{\emph{\glshypernumber{#1}}}

```

1.16 Acronyms

```

\oldacronym \oldacronym[label]{abbrv}{long}{key-val list}

```

This emulates the way the old package defined acronyms. It is equivalent to `\newacronym[key-val list]{label}{abbrv}{long}` and it additionally defines the command `\<label>` which is equivalent to `\gls{<label>}` (thus `<label>` must only contain alphabetical characters). If `<label>` is omitted, `<abbrv>` is used. This only emulates the syntax of the old package. The way the acronyms appear in the list of acronyms is determined by the definition of `\newacronym` and the glossary style.

Note that `\<label>` can't have an optional argument if the package is loaded. If hasn't been loaded then you can do `\<label>[<insert>]` but you can't do `\<label>[<key-val list>]`. For example if you define the acronym `svm`, then you can do `\svm['s]` but you can't do `\svm[format=textbf]`. If the package is loaded, `\svm['s]` will appear as `svm ['s]` which is unlikely to be the desired result. In this case, you will need to use `\gls` explicitly, e.g. `\gls{svm}['s]`. Note that it is up to the user to load if desired.

```

5624 \newcommand{\oldacronym}[4][\gls@label]{%
5625   \def\gls@label{#2}%
5626   \newacronym[#4]{#1}{#2}{#3}%
5627   \ifcsundef{xspace}%
5628     {%

```



```

5629 \expandafter\edef\csname#1\endcsname{%
5630 \noexpand\@ifstar{\noexpand\Gls{#1}}{\noexpand\gls{#1}}}%
5631 }%
5632 }%
5633 {%
5634 \expandafter\edef\csname#1\endcsname{%
5635 \noexpand\@ifstar{\noexpand\Gls{#1}\noexpand\xspace}{%
5636 \noexpand\gls{#1}\noexpand\xspace}%
5637 }%
5638 }%
5639 }

```

`\newacronym[⟨key-val list⟩]{⟨label⟩}{⟨abbrev⟩}{⟨long⟩}`

This is a quick way of defining acronyms, using `\newglossaryentry` with the appropriate values. It sets the glossary type to `\acronymtype` which will be acronym if the package option `acronym` has been used, otherwise it will be the default glossary. Since `\newacronym` merely calls `\newglossaryentry`, the acronym is treated like any other glossary entry.

If you prefer a different format, you can redefine `\newacronym` as required. The optional argument can be used to override any of the settings.

This is just a stub. It's redefined by commands like `\SetDefaultAcronymStyle`.

`\newacronym`

```

5640 \newcommand{\newacronym}[4] [] {}

```

Set up some convenient short cuts. These need to be changed if `\newacronym` is changed (or if the description key is changed).

`\acrpluralsuffix` Plural suffix used by `\newacronym`. This just defaults to `\glspluralsuffix` but is changed to include `\textup` if the `smallcaps` option is used, so that the suffix doesn't appear in small caps as it doesn't look right. For example, `ABCS` looks as though the “s” is part of the acronym, but `ABCs` looks as though the “s” is a plural suffix. Since the entire text `abcs` is set in `\textsc`, `\textup` is needed to cancel it out.

```

5641 \newcommand*{\acrpluralsuffix}{\glspluralsuffix}

```

If `garamondx` has been loaded, need to use `\textulc` instead of `\textup`.

`\glstextup`

```

5642 \newrobustcmd*{\glstextup}[1]{\ifdef\textulc{\textulc{#1}}{\textup{#1}}}

```

The following are defined for compatibility with version 2.07 and earlier.

`\glsshortkey`

```

5643 \newcommand*{\glsshortkey}{short}

```

```

\glsshortpluralkey
5644 \newcommand*{\glsshortpluralkey}{shortplural}

\glslongkey
5645 \newcommand*{\glslongkey}{long}

\glslongpluralkey
5646 \newcommand*{\glslongpluralkey}{longplural}

\acrfull  Full form of the acronym.
5647 \newrobustcmd*{\acrfull}{\@gls@hyp@opt\ns@acrfull}

5648 \newcommand*\ns@acrfull[2][\%
5649   \new@ifnextchar[\@acrfull{#1}{#2}]{\%
5650     {\@acrfull{#1}{#2}[]}\%
5651 }

\@acrfull  Low-level macro:
5652 \def\@acrfull#1#2[#3]{\%
    Make it easier for acronym styles to change this:
5653   \acrfullfmt{#1}{#2}{#3}%
5654 }

    Using \acrlinkfullformat and \acrfullformat is now deprecated as it
    can cause complications with the first letter upper case variants, but the pack-
    age needs to provide backward compatibility support.

\acrfullfmt  No case change full format.
5655 \newcommand*{\acrfullfmt}[3]{\%
5656   \acrlinkfullformat{\@acrlong}{\@acrshort}{#1}{#2}{#3}%
5657 }

\acrlinkfullformat  Format for full links like \acrfull. Syntax: \acrlinkfullformat{<long
cs>}{<short cs>}{<options>}{<label>}{<insert>}
5658 \newcommand{\acrlinkfullformat}[5]{\%
5659   \acrfullformat{#1{#3}{#4}[#5]}{#2{#3}{#4}[]}\%
5660 }

\acrfullformat  Default full form is <long> (<short>).
5661 \newcommand{\acrfullformat}[2]{#1\glsspace(#2)}

\glsspace  Robust space to ensure it's written to the .glsdefs file.
5662 \newrobustcmd{\glsspace}{\space}

    Default format for full acronym

\Acrfull
5663 \newrobustcmd*{\Acrfull}{\@gls@hyp@opt\ns@Acrfull}

```

```

5664 \newcommand*\ns@Acrfull[2][]{%
5665   \new@ifnextchar[{\@Acrfull{#1}{#2}}{%
5666     {\@Acrfull{#1}{#2}}[]}%
5667 }

```

Low-level macro:

```

5668 \def\@Acrfull#1#2[#3]{%

```

Make it easier for acronym styles to change this:

```

5669   \Acrfullfmt{#1}{#2}{#3}%
5670 }

```

`\Acrfullfmt` First letter upper case full format.

```

5671 \newcommand*\@Acrfullfmt[3]{%
5672   \acrlinkfullformat{\@Acrlong}{\@acrshort}{#1}{#2}{#3}%
5673 }

```

`\ACRfull`

```

5674 \newrobustcmd*\@ACRfull{\@gls@hyp@opt\ns@ACRfull}

```

```

5675 \newcommand*\ns@ACRfull[2][]{%
5676   \new@ifnextchar[{\@ACRfull{#1}{#2}}{%
5677     {\@ACRfull{#1}{#2}}[]}%
5678 }

```

Low-level macro:

```

5679 \def\@ACRfull#1#2[#3]{%

```

Make it easier for acronym styles to change this:

```

5680   \ACRfullfmt{#1}{#2}{#3}%
5681 }

```

`\ACRfullfmt` All upper case full format.

```

5682 \newcommand*\@ACRfullfmt[3]{%
5683   \acrlinkfullformat{\@ACRlong}{\@ACRshort}{#1}{#2}{#3}%
5684 }

```

Plural:

`\acrfullpl`

```

5685 \newrobustcmd*\@acrfullpl{\@gls@hyp@opt\ns@acrfullpl}

```

```

5686 \newcommand*\ns@acrfullpl[2][]{%
5687   \new@ifnextchar[{\@acrfullpl{#1}{#2}}{%
5688     {\@acrfullpl{#1}{#2}}[]}%
5689 }

```

Low-level macro:

```

5690 \def\@acrfullpl#1#2[#3]{%

```

Make it easier for acronym styles to change this:

```
5691 \acrfullplfmt{#1}{#2}{#3}%  
5692 }
```

\acrfullplfmt No case change plural full format.

```
5693 \newcommand*\acrfullplfmt[3]{%  
5694 \acrlinkfullformat{\@acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%  
5695 }
```

\Acrfullpl

```
5696 \newrobustcmd*\Acrfullpl{\@gls@hyp@opt\ns@Acrfullpl}  
  
5697 \newcommand*\ns@Acrfullpl[2][]{%  
5698 \new@ifnextchar[{\@Acrfullpl{#1}{#2}}%  
5699 {\@Acrfullpl{#1}{#2}[]}%  
5700 }
```

Low-level macro:

```
5701 \def\@Acrfullpl#1#2[#3]{%
```

Make it easier for acronym styles to change this:

```
5702 \Acrfullplfmt{#1}{#2}{#3}%  
5703 }
```

\Acrfullplfmt First letter upper case plural full format.

```
5704 \newcommand*\Acrfullplfmt[3]{%  
5705 \acrlinkfullformat{\@Acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%  
5706 }
```

\ACRfullpl

```
5707 \newrobustcmd*\ACRfullpl{\@gls@hyp@opt\ns@ACRfullpl}  
  
5708 \newcommand*\ns@ACRfullpl[2][]{%  
5709 \new@ifnextchar[{\@ACRfullpl{#1}{#2}}%  
5710 {\@ACRfullpl{#1}{#2}[]}%  
5711 }
```

Low-level macro:

```
5712 \def\@ACRfullpl#1#2[#3]{%
```

Make it easier for acronym styles to change this:

```
5713 \ACRfullplfmt{#1}{#2}{#3}%  
5714 }
```

\ACRfullplfmt All upper case plural full format.

```
5715 \newcommand*\ACRfullplfmt[3]{%  
5716 \acrlinkfullformat{\@ACRlongpl}{\@ACRshortpl}{#1}{#2}{#3}%  
5717 }
```

1.17 Predefined acronym styles

`\acronymfont` This is only used with the additional acronym styles:

```
5718 \newcommand{\acronymfont}[1]{#1}
```

`\firstacronymfont` This is only used with the additional acronym styles:

```
5719 \newcommand{\firstacronymfont}[1]{\acronymfont{#1}}
```

`\acronymformat` The styles that allow an additional description use `\acronymformat{<short>}{<long>}` to determine what information is displayed in the name.

```
5720 \newcommand*{\acronymformat}[2]{\acronymfont{#1}}
```

Define some tokens used by `\newacronym`:

`\glskeylisttok`

```
5721 \newtoks\glskeylisttok
```

`\glslabeltok`

```
5722 \newtoks\glslabeltok
```

`\glsshorttok`

```
5723 \newtoks\glsshorttok
```

`\glslongtok`

```
5724 \newtoks\glslongtok
```

`\newacronymhook` Provide a hook for `\newacronym`:

```
5725 \newcommand*{\newacronymhook}{}
```

`\setGenericNewAcronym` New improved version of setting the acronym style.

```
5726 \newcommand*{\SetGenericNewAcronym}{%
5727   \renewcommand{\newacronym}[4][[]]{%
5728     \ifdefempty{\@glsacronymlists}%
5729     {%
5730       \def\@glo@type{\acronymtype}%
5731       \setkeys{glossentry}{##1}%
5732       \DeclareAcronymList{\@glo@type}%
5733     }%
5734   }%
5735   \glskeylisttok{##1}%
5736   \glslabeltok{##2}%
5737   \glsshorttok{##3}%
5738   \glslongtok{##4}%
5739   \newacronymhook
5740   \protected@edef\@do@newglossaryentry{%
5741     \noexpand\newglossaryentry{\the\glslabeltok}%
5742     {%
5743       type=\acronymtype,%
```

```

5744     name={\expandonce{\acronymentry{##2}}},%
5745     sort={\acronymsort{\the\glsshorttok}{\the\glslongtok}},%
5746     text={\the\glsshorttok},%
5747     short={\the\glsshorttok},%
5748     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5749     long={\the\glslongtok},%
5750     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5751     \GenericAcronymFields,%
5752     \the\glskeylisttok
5753   }%
5754 }%
5755 \do@newglossaryentry
5756 }%

```

Make sure that \acrfull etc reflects the new style:

```

5757 \renewcommand*{\acrfullfmt}[3]{%
5758   \glslink[##1]{##2}{\genacrfullformat{##2}{##3}}}%
5759 \renewcommand*{\Acrfullfmt}[3]{%
5760   \glslink[##1]{##2}{\Genacrfullformat{##2}{##3}}}%
5761 \renewcommand*{\ACRfullfmt}[3]{%
5762   \glslink[##1]{##2}{%
5763     \mfirstucMakeUppercase{\genacrfullformat{##2}{##3}}}%
5764 \renewcommand*{\acrfullplfmt}[3]{%
5765   \glslink[##1]{##2}{\genplacrfullformat{##2}{##3}}}%
5766 \renewcommand*{\Acrfullplfmt}[3]{%
5767   \glslink[##1]{##2}{\Genplacrfullformat{##2}{##3}}}%
5768 \renewcommand*{\ACRfullplfmt}[3]{%
5769   \glslink[##1]{##2}{%
5770     \mfirstucMakeUppercase{\genplacrfullformat{##2}{##3}}}%

```

Make sure that \glsentryfull etc reflects the new style:

```

5771 \renewcommand*{\glsentryfull}[1]{\genacrfullformat{##1}{}}%
5772 \renewcommand*{\Glsentryfull}[1]{\Genacrfullformat{##1}{}}%
5773 \renewcommand*{\glsentryfullpl}[1]{\genplacrfullformat{##1}{}}%
5774 \renewcommand*{\Glsentryfullpl}[1]{\Genplacrfullformat{##1}{}}%
5775 }

```

`\GenericAcronymFields` Fields used by `\SetGenericNewAcronym` that can be changed by the acronym style.

```

5776 \newcommand*{\GenericAcronymFields}{description={\the\glslongtok}}

```

`\acronymentry` `\acronymentry{<label>}`

Display style for the name field in the list of acronyms.

```

5777 \newcommand*{\acronymentry}[1]{\acronymfont{\glsentryshort{#1}}}

```

`\acronymsort` `\acronymsort{<short>}{<long>}`

Default sort format for acronyms.

```
5778 \newcommand*{\acronymsort}[2]{#1}
```

`\setacronymstyle` `\setacronymstyle{<style name>}`

```
5779 \newcommand*{\setacronymstyle}[1]{%
5780   \ifcsundef{@glsacr@dispstyle@#1}%
5781   {%
5782     \PackageError{glossaries}{Undefined acronym style ‘#1’}{}%
5783   }%
5784   {%
5785     \ifdefempty{\@glsacronymlists}%
5786     {%
5787       \DeclareAcronymList{<acronym type>%
5788     }%
5789     {}%
5790     \SetGenericNewAcronym
5791     \GlsUseAcrStyleDefs{#1}%
5792     \@for\@gls@type:=\@glsacronymlists\do{%
5793       \defglsentryfmt[\@gls@type]{\GlsUseAcrEntryDispStyle{#1}}%
5794     }%
5795   }%
5796 }
```

`\newacronymstyle` `\newacronymstyle{<style name>}{<entry format definition>}{<display definitions>}`

Defines a new acronym style called <style name>.

```
5797 \newcommand*{\newacronymstyle}[3]{%
5798   \ifcsdef{@glsacr@dispstyle@#1}%
5799   {%
5800     \PackageError{glossaries}{Acronym style ‘#1’ already exists}{}%
5801   }%
5802   {%
5803     \csdef{@glsacr@dispstyle@#1}{#2}%
5804     \csdef{@glsacr@styledefs@#1}{#3}%
5805   }%
5806 }
```

`\renewacronymstyle` Redefines the given acronym style.

```
5807 \newcommand*{\renewacronymstyle}[3]{%
5808   \ifcsdef{@glsacr@dispstyle@#1}%
5809   {%
5810     \csdef{@glsacr@dispstyle@#1}{#2}%
5811     \csdef{@glsacr@styledefs@#1}{#3}%
5812   }%
```

```

5813  {%
5814    \PackageError{glossaries}{Acronym style ‘#1’ doesn’t exist}{}%
5815  }%
5816 }

```

seAcrEntryDispStyle

```

5817 \newcommand*{\GlsUseAcrEntryDispStyle}[1]{\csuse{@glsacr@dispstyle@#1}}

```

\GlsUseAcrStyleDefs

```

5818 \newcommand*{\GlsUseAcrStyleDefs}[1]{\csuse{@glsacr@styledefs@#1}}

```

Predefined acronym styles:

long-short *⟨long⟩* (*⟨short⟩*) acronym style.

```

5819 \newacronymstyle{long-short}%
5820 {%

```

Check for long form in case this is a mixed glossary.

```

5821  \ifglshaslong{\glslabel}{\glsgenacfmt}{\glsgenentryfmt}%
5822 }%
5823 {%
5824  \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
5825  \renewcommand*{\genacrfullformat}[2]{%
5826    \glsentrylong{##1}##2\space
5827    (\protect\firstacronymfont{\glsentryshort{##1}})%
5828  }%
5829  \renewcommand*{\Genacrfullformat}[2]{%
5830    \Glsentrylong{##1}##2\space
5831    (\protect\firstacronymfont{\glsentryshort{##1}})%
5832  }%
5833  \renewcommand*{\genplacrfullformat}[2]{%
5834    \glsentrylongpl{##1}##2\space
5835    (\protect\firstacronymfont{\glsentryshortpl{##1}})%
5836  }%
5837  \renewcommand*{\Genplacrfullformat}[2]{%
5838    \Glsentrylongpl{##1}##2\space
5839    (\protect\firstacronymfont{\glsentryshortpl{##1}})%
5840  }%
5841  \renewcommand*{\acronymentry}[1]{\acronymfont{\glsentryshort{##1}}}
5842  \renewcommand*{\acronymsort}[2]{##1}%
5843  \renewcommand*{\acronymfont}[1]{##1}%
5844  \renewcommand*{\firstacronymfont}[1]{\acronymfont{##1}}%
5845  \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
5846 }

```

short-long *⟨short⟩* (*⟨long⟩*) acronym style.

```

5847 \newacronymstyle{short-long}%
5848 {%

```


Check for long form in case this is a mixed glossary.

```

5849 \ifglshaslong{\glslabel}{\glsngenacfmt}{\glsngenentryfmt}%
5850 }%
5851 {%
5852 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
5853 \renewcommand*{\genacrfullformat}[2]{%
5854 \protect\firstacronymfont{\glsentryshort{##1}}##2\space
5855 (\glsentrylong{##1})%
5856 }%
5857 \renewcommand*{\Genacrfullformat}[2]{%
5858 \protect\firstacronymfont{\Glsentryshort{##1}}##2\space
5859 (\glsentrylong{##1})%
5860 }%
5861 \renewcommand*{\genplacrfullformat}[2]{%
5862 \protect\firstacronymfont{\glsentryshortpl{##1}}##2\space
5863 (\glsentrylongpl{##1})%
5864 }%
5865 \renewcommand*{\Genplacrfullformat}[2]{%
5866 \protect\firstacronymfont{\Glsentryshortpl{##1}}##2\space
5867 (\glsentrylongpl{##1})%
5868 }%
5869 \renewcommand*{\acronymentry}[1]{\acronymfont{\glsentryshort{##1}}}%
5870 \renewcommand*{\acronymsort}[2]{##1}%
5871 \renewcommand*{\acronymfont}[1]{##1}%
5872 \renewcommand*{\firstacronymfont}[1]{\acronymfont{##1}}%
5873 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
5874 }

```

long-sc-short *<long>* (\textsc{<short>}) acronym style.

```

5875 \newacronymstyle{long-sc-short}%
5876 {%
5877 \GlsUseAcrEntryDisplayStyle{long-short}%
5878 }%
5879 {%
5880 \GlsUseAcrStyleDefs{long-short}%
5881 \renewcommand{\acronymfont}[1]{\textsc{##1}}%
5882 \renewcommand*{\acrpluralsuffix}{\glstextup{\glspluralsuffix}}%
5883 }

```

long-sm-short *<long>* (\textsmaller{<short>}) acronym style.

```

5884 \newacronymstyle{long-sm-short}%
5885 {%
5886 \GlsUseAcrEntryDisplayStyle{long-short}%
5887 }%
5888 {%
5889 \GlsUseAcrStyleDefs{long-short}%
5890 \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
5891 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
5892 }

```

sc-short-long *<short>* (\textsc{<long>}) acronym style.

```

5893 \newacronymstyle{sc-short-long}%
5894 {%
5895   \GlsUseAcrEntryDisplayStyle{short-long}%
5896 }%
5897 {%
5898   \GlsUseAcrStyleDefs{short-long}%
5899   \renewcommand{\acronymfont}[1]{\textsc{##1}}%
5900   \renewcommand*{\acrpluralsuffix}{\glstextup{\glspluralsuffix}}%
5901 }
```

sm-short-long *<short>* (\textsmaller{<long>}) acronym style.

```

5902 \newacronymstyle{sm-short-long}%
5903 {%
5904   \GlsUseAcrEntryDisplayStyle{short-long}%
5905 }%
5906 {%
5907   \GlsUseAcrStyleDefs{short-long}%
5908   \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
5909   \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
5910 }
```

long-short-desc *<long>* ({<short>}) acronym style that has an accompanying description (which the user needs to supply).

```

5911 \newacronymstyle{long-short-desc}%
5912 {%
5913   \GlsUseAcrEntryDisplayStyle{long-short}%
5914 }%
5915 {%
5916   \GlsUseAcrStyleDefs{long-short}%
5917   \renewcommand*{\GenericAcronymFields}{}%
5918   \renewcommand*{\acronymsort}[2]{##2}%
5919   \renewcommand*{\acronymentry}[1]{%
5920     \glsentrylong{##1}\space (\acronymfont{\glsentryshort{##1}})}%
5921 }
```

long-sc-short-desc *<long>* (\textsc{<short>}) acronym style that has an accompanying description (which the user needs to supply).

```

5922 \newacronymstyle{long-sc-short-desc}%
5923 {%
5924   \GlsUseAcrEntryDisplayStyle{long-sc-short}%
5925 }%
5926 {%
5927   \GlsUseAcrStyleDefs{long-sc-short}%
5928   \renewcommand*{\GenericAcronymFields}{}%
5929   \renewcommand*{\acronymsort}[2]{##2}%
5930   \renewcommand*{\acronymentry}[1]{%
5931     \glsentrylong{##1}\space (\textsc{\acronymfont{\glsentryshort{##1}})}%
5932 }
```

long-sm-short-desc *⟨long⟩* (\textsmaller{⟨short⟩}) acronym style that has an accompanying description (which the user needs to supply).

```
5933 \newacronymstyle{long-sm-short-desc}%
5934 {%
5935   \GlsUseAcrEntryDispStyle{long-sm-short}%
5936 }%
5937 {%
5938   \GlsUseAcrStyleDefs{long-sm-short}%
5939   \renewcommand*{\GenericAcronymFields}{}%
5940   \renewcommand*{\acronymsort}[2]{##2}%
5941   \renewcommand*{\acronymentry}[1]{%
5942     \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
5943 }
```

short-long-desc *⟨short⟩* ({⟨long⟩}) acronym style that has an accompanying description (which the user needs to supply).

```
5944 \newacronymstyle{short-long-desc}%
5945 {%
5946   \GlsUseAcrEntryDispStyle{short-long}%
5947 }%
5948 {%
5949   \GlsUseAcrStyleDefs{short-long}%
5950   \renewcommand*{\GenericAcronymFields}{}%
5951   \renewcommand*{\acronymsort}[2]{##2}%
5952   \renewcommand*{\acronymentry}[1]{%
5953     \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
5954 }
```

sc-short-long-desc *⟨long⟩* (\textsc{⟨short⟩}) acronym style that has an accompanying description (which the user needs to supply).

```
5955 \newacronymstyle{sc-short-long-desc}%
5956 {%
5957   \GlsUseAcrEntryDispStyle{sc-short-long}%
5958 }%
5959 {%
5960   \GlsUseAcrStyleDefs{sc-short-long}%
5961   \renewcommand*{\GenericAcronymFields}{}%
5962   \renewcommand*{\acronymsort}[2]{##2}%
5963   \renewcommand*{\acronymentry}[1]{%
5964     \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
5965 }
```

sm-short-long-desc *⟨long⟩* (\textsmaller{⟨short⟩}) acronym style that has an accompanying description (which the user needs to supply).

```
5966 \newacronymstyle{sm-short-long-desc}%
5967 {%
5968   \GlsUseAcrEntryDispStyle{sm-short-long}%
5969 }%
```

```

5970 {%
5971   \GlsUseAcrStyleDefs{sm-short-long}%
5972   \renewcommand*{\GenericAcronymFields}{}%
5973   \renewcommand*{\acronymsort}[2]{##2}%
5974   \renewcommand*{\acronymentry}[1]{%
5975     \glstrylong{##1}\space (\acronymfont{\glstryshort{##1}})}%
5976 }

```

dua *<long>* only acronym style.

```

5977 \newacronymstyle{dua}%
5978 {%

```

Check for long form in case this is a mixed glossary.

```

5979   \ifdefempty\glscustomtext
5980   {%
5981     \ifglshaslong{\glslabel}%
5982     {%
5983       \glssifplural
5984       {%

```

Plural form:

```

5985       \glscapscase
5986       {%

```

Plural form, don't adjust case:

```

5987       \glstrylongpl{\glslabel}\glssinsert
5988       }%
5989       {%

```

Plural form, make first letter upper case:

```

5990       \Glsentrylongpl{\glslabel}\glssinsert
5991       }%
5992       {%

```

Plural form, all caps:

```

5993       \mfirstucMakeUppercase
5994       {\glstrylongpl{\glslabel}\glssinsert}%
5995       }%
5996       }%
5997       {%

```

Singular form

```

5998       \glscapscase
5999       {%

```

Singular form, don't adjust case:

```

6000       \glstrylong{\glslabel}\glssinsert
6001       }%
6002       {%

```

Subsequent singular form, make first letter upper case:

```

6003       \Glsentrylong{\glslabel}\glssinsert

```

```

6004         }%
6005         {%

Subsequent singular form, all caps:
6006         \mfirstucMakeUppercase
6007         {\glsentrylong{\glslabel}\glsinsert}%
6008         }%
6009         }%
6010         }%
6011         {%

Not an acronym:
6012         \glsgenentryfmt
6013         }%
6014         }%
6015         {\glscustomtext\glsinsert}%
6016 }%
6017 {%
6018 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%

6019 \renewcommand*{\acrfullfmt}[3]{%
6020 \glslink[##1]{##2}{\glsentrylong{##2}##3\space
6021 (\acronymfont{\glsentryshort{##2}})}}%
6022 \renewcommand*{\Acrfullfmt}[3]{%
6023 \glslink[##1]{##2}{\Glsentrylong{##2}##3\space
6024 (\acronymfont{\glsentryshort{##2}})}}%
6025 \renewcommand*{\ACRfullfmt}[3]{%
6026 \glslink[##1]{##2}{%
6027 \mfirstucMakeUppercase{\glsentrylong{##2}##3\space
6028 (\acronymfont{\glsentryshort{##2}})}}}%

6029 \renewcommand*{\acrfullplfmt}[3]{%
6030 \glslink[##1]{##2}{\glsentrylongpl{##2}##3\space
6031 (\acronymfont{\glsentryshortpl{##2}})}}%

6032 \renewcommand*{\Acrfullplfmt}[3]{%
6033 \glslink[##1]{##2}{\Glsentrylongpl{##2}##3\space
6034 (\acronymfont{\glsentryshortpl{##2}})}}%
6035 \renewcommand*{\ACRfullplfmt}[3]{%
6036 \glslink[##1]{##2}{%
6037 \mfirstucMakeUppercase{\glsentrylongpl{##2}##3\space
6038 (\acronymfont{\glsentryshortpl{##2}})}}}%
6039 \renewcommand*{\glsentryfull}[1]{%
6040 \glsentrylong{##1}\space(\acronymfont{\glsentryshort{##1}})%
6041 }%
6042 \renewcommand*{\Glsentryfull}[1]{%
6043 \Glsentrylong{##1}\space(\acronymfont{\glsentryshort{##1}})%
6044 }%
6045 \renewcommand*{\glsentryfullpl}[1]{%
6046 \glsentrylongpl{##1}\space(\acronymfont{\glsentryshortpl{##1}})%
6047 }%

```

```

6048 \renewcommand*\Glsentryfullpl}[1]{%
6049 \Glsentrylongpl{##1}\space(\acronymfont{\Glsentryshortpl{##1}})%
6050 }%
6051 \renewcommand*\acronymentry}[1]{\acronymfont{\Glsentryshort{##1}}}%
6052 \renewcommand*\acronymsort}[2]{##1}%
6053 \renewcommand*\acronymfont}[1]{##1}%
6054 \renewcommand*\acrpluralsuffix{\Glspluralsuffix}%
6055 }

```

dua-desc *<long>* only acronym style with user-supplied description.

```

6056 \newacronymstyle{dua-desc}%
6057 {%
6058 \GlsUseAcrEntryDispStyle{dua}%
6059 }%
6060 {%
6061 \GlsUseAcrStyleDefs{dua}%
6062 \renewcommand*\GenericAcronymFields{}}%

6063 \renewcommand*\acronymentry}[1]{\acronymfont{\Glsentrylong{##1}}}%
6064 \renewcommand*\acronymsort}[2]{##2}%
6065 }%

```

footnote *<short>*\footnote{*<long>*} acronym style.

```

6066 \newacronymstyle{footnote}%
6067 {%

```

Check for long form in case this is a mixed glossary.

```

6068 \ifglshaslong{\Glslabel}{\GlsGenacFmt}{\GlsGenentryFmt}%
6069 }%
6070 {%
6071 \renewcommand*\GenericAcronymFields{description={\the\GlsLongTok}}%

```

Need to ensure hyperlinks are switched off on first use:

```

6072 \glshyperfirstfalse
6073 \renewcommand*\GenacrFullFormat}[2]{%
6074 \protect\firstacronymfont{\Glsentryshort{##1}}##2%
6075 \protect\footnote{\Glsentrylong{##1}}%
6076 }%
6077 \renewcommand*\GenacrFullFormat}[2]{%
6078 \firstacronymfont{\Glsentryshort{##1}}##2%
6079 \protect\footnote{\Glsentrylong{##1}}%
6080 }%
6081 \renewcommand*\GenplacrFullFormat}[2]{%
6082 \protect\firstacronymfont{\Glsentryshortpl{##1}}##2%
6083 \protect\footnote{\Glsentrylongpl{##1}}%
6084 }%
6085 \renewcommand*\GenplacrFullFormat}[2]{%
6086 \protect\firstacronymfont{\Glsentryshortpl{##1}}##2%
6087 \protect\footnote{\Glsentrylongpl{##1}}%
6088 }%

```

```

6089 \renewcommand*{\acronymentry}[1]{\acronymfont{\glentryshort{##1}}}%
6090 \renewcommand*{\acronymsort}[2]{##1}%
6091 \renewcommand*{\acronymfont}[1]{##1}%
6092 \renewcommand*{\acrpluralsuffix}{\glpluralsuffix}%

```

Don't use footnotes for \acrfull:

```

6093 \renewcommand*{\acrfullfmt}[3]{%
6094   \glslink[##1]{##2}{\acronymfont{\glentryshort{##2}}##3\space
6095   (\glentrylong{##2})}%
6096 \renewcommand*{\Acrfullfmt}[3]{%
6097   \glslink[##1]{##2}{\acronymfont{\Glentryshort{##2}}##3\space
6098   (\glentrylong{##2})}%
6099 \renewcommand*{\ACRfullfmt}[3]{%
6100   \glslink[##1]{##2}{%
6101     \mfirstucMakeUppercase{\acronymfont{\glentryshort{##2}}##3\space
6102     (\glentrylong{##2})}}}%
6103 \renewcommand*{\acrfullplfmt}[3]{%
6104   \glslink[##1]{##2}{\acronymfont{\glentryshortpl{##2}}##3\space
6105   (\glentrylongpl{##2})}%
6106 \renewcommand*{\Acrfullplfmt}[3]{%
6107   \glslink[##1]{##2}{\acronymfont{\Glentryshortpl{##2}}##3\space
6108   (\glentrylongpl{##2})}%
6109 \renewcommand*{\ACRfullplfmt}[3]{%
6110   \glslink[##1]{##2}{%
6111     \mfirstucMakeUppercase{\acronymfont{\glentryshortpl{##2}}##3\space
6112     (\glentrylongpl{##2})}}}%

```

Similarly for \glentryfull etc:

```

6113 \renewcommand*{\glentryfull}[1]{%
6114   \acronymfont{\glentryshort{##1}}\space(\glentrylong{##1})}%
6115 \renewcommand*{\Glentryfull}[1]{%
6116   \acronymfont{\Glentryshort{##1}}\space(\glentrylong{##1})}%
6117 \renewcommand*{\glentryfullpl}[1]{%
6118   \acronymfont{\glentryshortpl{##1}}\space(\glentrylongpl{##1})}%
6119 \renewcommand*{\Glentryfullpl}[1]{%
6120   \acronymfont{\Glentryshortpl{##1}}\space(\glentrylongpl{##1})}%
6121 }

```

footnote-sc \textsc{<short>}\footnote{<long>} acronym style.

```

6122 \newacronymstyle{footnote-sc}%
6123 {%
6124   \GlsUseAcrEntryDispStyle{footnote}%
6125 }%
6126 {%
6127   \GlsUseAcrStyleDefs{footnote}%
6128   \renewcommand{\acronymentry}[1]{\acronymfont{\glentryshort{##1}}}
6129   \renewcommand{\acronymfont}[1]{\textsc{##1}}%
6130   \renewcommand*{\acrpluralsuffix}{\glstextup{\glpluralsuffix}}%
6131 }%

```

footnote-sm \textsmaller{<short>}\footnote{<long>} acronym style.

```
6132 \newacronymstyle{footnote-sm}%
6133 {%
6134   \GlsUseAcrEntryDispStyle{footnote}%
6135 }%
6136 {%
6137   \GlsUseAcrStyleDefs{footnote}%
6138   \renewcommand{\acronymentry}[1]{\acronymfont{\glsentryshort{##1}}}
6139   \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
6140   \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
6141 }%
```

footnote-desc <short>\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```
6142 \newacronymstyle{footnote-desc}%
6143 {%
6144   \GlsUseAcrEntryDispStyle{footnote}%
6145 }%
6146 {%
6147   \GlsUseAcrStyleDefs{footnote}%
6148   \renewcommand*{\GenericAcronymFields}{}%
6149   \renewcommand*{\acronymsort}[2]{##2}%
6150   \renewcommand*{\acronymentry}[1]{%
6151     \glsentrylong{##1}\space (\acronymfont{\glsentryshort{##1}})}%
6152 }
```

footnote-sc-desc \textsc{<short>}\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```
6153 \newacronymstyle{footnote-sc-desc}%
6154 {%
6155   \GlsUseAcrEntryDispStyle{footnote-sc}%
6156 }%
6157 {%
6158   \GlsUseAcrStyleDefs{footnote-sc}%
6159   \renewcommand*{\GenericAcronymFields}{}%
6160   \renewcommand*{\acronymsort}[2]{##2}%
6161   \renewcommand*{\acronymentry}[1]{%
6162     \glsentrylong{##1}\space (\acronymfont{\glsentryshort{##1}})}%
6163 }
```

footnote-sm-desc \textsmaller{<short>}\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```
6164 \newacronymstyle{footnote-sm-desc}%
6165 {%
6166   \GlsUseAcrEntryDispStyle{footnote-sm}%
6167 }%
6168 {%
6169   \GlsUseAcrStyleDefs{footnote-sm}%

```



```

6170 \renewcommand*{\GenericAcronymFields}{}%
6171 \renewcommand*{\acronymsort}[2]{##2}%
6172 \renewcommand*{\acronymentry}[1]{%
6173     \glsentrylong{##1}\space (\acronymfont{\glsentryshort{##1}})}%
6174 }

```

`\fineAcronymSynonyms`

```

6175 \newcommand*{\DefineAcronymSynonyms}{%

```

Short form

`\acs`

```

6176 \let\acs\acrshort

```

First letter uppercase short form

`\Acs`

```

6177 \let\Acs\Acrshort

```

Plural short form

`\acsp`

```

6178 \let\acsp\acrshortpl

```

First letter uppercase plural short form

`\Acsp`

```

6179 \let\Acsp\Acrshortpl

```

Long form

`\acl`

```

6180 \let\acl\aclong

```

Plural long form

`\aclp`

```

6181 \let\aclp\aclongpl

```

First letter upper case long form

`\Acl`

```

6182 \let\Acl\Aclong

```

First letter upper case plural long form

`\Aclp`

```

6183 \let\Aclp\Aclongpl

```

Full form

`\acf`

```

6184 \let\acf\acrfull

```

Plural full form

`\acfp`

```
6185 \let\acfp\acrfullpl
```

First letter upper case full form

`\Acf`

```
6186 \let\Acf\Acrfull
```

First letter upper case plural full form

`\Acfp`

```
6187 \let\Acfp\Acrfullpl
```

Standard form

`\ac`

```
6188 \let\ac\gls
```

First upper case standard form

`\Ac`

```
6189 \let\Ac\Gls
```

Standard plural form

`\acp`

```
6190 \let\acp\glspl
```

Standard first letter upper case plural form

`\Acp`

```
6191 \let\Acp\Glspl
```

```
6192 }
```

Define synonyms if required

```
6193 \ifglsacrshortcuts
```

```
6194 \DefineAcronymSynonyms
```

```
6195 \fi
```

These commands for setting the style are now deprecated but are kept for backward compatibility.

AcronymDisplayStyle Sets the default acronym display style for given glossary.

```
6196 \newcommand*\SetDefaultAcronymDisplayStyle}[1]{%
```

```
6197 \defglsentryfmt[#1]{\glsentryfmt}%
```

```
6198 }
```

`\defaultNewAcronymDef` Sets up the acronym definition for the default style. The information is provided by the tokens `\glslabeltok`, `\glsshorttok`, `\glslongtok` and `\glskeylisttok`.

```

6199 \newcommand*{\DefaultNewAcronymDef}{%
6200   \edef\@do@newglossaryentry{%
6201     \noexpand\newglossaryentry{\the\glslabeltok}%
6202     {%
6203       type=\acronymtype,%
6204       name={\the\glsshorttok},%
6205       sort={\the\glsshorttok},%
6206       text={\the\glsshorttok},%
6207       first={\acrfullformat{\the\glslongtok}{\the\glsshorttok}},%
6208       plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6209       firstplural={\acrfullformat{\noexpand\expandonce\noexpand\@glo@longpl}%
6210                    {\noexpand\expandonce\noexpand\@glo@shortpl}},%
6211       short={\the\glsshorttok},%
6212       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6213       long={\the\glslongtok},%
6214       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6215       description={\the\glslongtok},%
6216       descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%

```

Remaining options specified by the user:

```

6217   \the\glskeylisttok
6218 }%
6219 }%
6220 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6221 \let\@org@gls@assign@plural\gls@assign@plural
6222 \let\@org@gls@assign@descplural\gls@assign@descplural
6223 \def\gls@assign@firstpl##1##2{%
6224   \@@gls@expand@field{##1}{firstpl}{##2}%
6225 }%
6226 \def\gls@assign@plural##1##2{%
6227   \@@gls@expand@field{##1}{plural}{##2}%
6228 }%
6229 \def\gls@assign@descplural##1##2{%
6230   \@@gls@expand@field{##1}{descplural}{##2}%
6231 }%
6232 \@do@newglossaryentry
6233 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6234 \let\gls@assign@plural\@org@gls@assign@plural
6235 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6236 }

```

`\DefaultAcronymStyle` Set up the default acronym style:

```

6237 \newcommand*{\SetDefaultAcronymStyle}{%
  Set the display style:
6238   \@for\@gls@type:=\@glsacronymlists\do{%
6239     \SetDefaultAcronymDisplayStyle{\@gls@type}%

```

6240 }%

Set up the definition of `\newacronym`:

6241 `\renewcommand{\newacronym}[4][\]{%`

If user is just using the main glossary and hasn't identified it as a list of acronyms, then update. (This is done to ensure backwards compatibility with versions prior to 2.04).

```
6242 \ifx\@glsacronymlists\@empty
6243 \def\@glo@type{\acronymtype}%
6244 \setkeys{glossentry}{##1}%
6245 \DeclareAcronymList{\@glo@type}%
6246 \SetDefaultAcronymDisplayStyle{\@glo@type}%
6247 \fi
6248 \glskeylisttok{##1}%
6249 \glslabeltok{##2}%
6250 \glsshorttok{##3}%
6251 \gslongtok{##4}%
6252 \newacronymhook
6253 \DefaultNewAcronymDef
6254 }%
6255 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
6256 }
```

`\acrfootnote` Used by the footnote acronym styles.

6257 `\newcommand*{\acrfootnote}[3]{\acrlinkfootnote{#1}{#2}{#3}}`

`\acrlinkfootnote`

```
6258 \newcommand*{\acrlinkfootnote}[3]{%
6259 \footnote{\glslink[#1]{#2}{#3}}%
6260 }
```

`\acrno linkfootnote`

```
6261 \newcommand*{\acrno linkfootnote}[3]{%
6262 \footnote{#3}%
6263 }
```

`AcronymDisplayStyle` Sets the acronym display style for given glossary for the description and footnote combination.

```
6264 \newcommand*{\SetDescriptionFootnoteAcronymDisplayStyle}[1]{%
6265 \def\glsentryfmt{#1}%

6266 \ifdefempty\glscustomtext
6267 {%
6268 \ifglsused{\glslabel}%
6269 {%
6270 \acronymfont{\glsentryfmt}%
6271 }%
6272 }
```

```

6273     \firstacronymfont{\glsgenentryfmt}%
6274     \ifglshassymbol{\glslabel}%
6275     {%
6276         \expandafter\protect\expandafter\acrfootnote\expandafter
6277         {\@gls@link@opts}{\@gls@link@label}%
6278         {%
6279             \glsifplural
6280             {\glsentrysymbolplural{\glslabel}}%
6281             {\glsentrysymbol{\glslabel}}%
6282         }%
6283     }%
6284 }%
6285 }%
6286 {\glscustomtext\glsinsert}%
6287 }%
6288 }

```

otnoteNewAcronymDef

```

6289 \newcommand*{\DescriptionFootnoteNewAcronymDef}{%
6290     \edef\@do@newglossaryentry{%
6291         \noexpand\newglossaryentry{\the\glslabeltok}%
6292         {%
6293             type=\acronymtype,%
6294             name={\noexpand\acronymfont{\the\glsshorttok}},%
6295             sort={\the\glsshorttok},%
6296             first={\the\glsshorttok},%
6297             firstplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6298             text={\the\glsshorttok},%
6299             plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6300             short={\the\glsshorttok},%
6301             shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6302             long={\the\glslongtok},%
6303             longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6304             symbol={\the\glslongtok},%
6305             symbolplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6306             \the\glskeylisttok
6307         }%
6308     }%
6309     \let\@org@gls@assign@firstpl\gls@assign@firstpl
6310     \let\@org@gls@assign@plural\gls@assign@plural
6311     \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6312     \def\gls@assign@firstpl##1##2{%
6313         \@@gls@expand@field{##1}{firstpl}{##2}%
6314     }%
6315     \def\gls@assign@plural##1##2{%
6316         \@@gls@expand@field{##1}{plural}{##2}%
6317     }%
6318     \def\gls@assign@symbolplural##1##2{%
6319         \@@gls@expand@field{##1}{symbolplural}{##2}%

```

```

6320 }%
6321 \@do@newglossaryentry
6322 \let\gls@assign@plural\@org@gls@assign@plural
6323 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6324 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6325 }

```

`ootnoteAcronymStyle` If a description and footnote are both required, store the long form in the symbol key. Store the short form in text key. Note that since the long form is stored in the symbol key, if you want the long form to appear in the list of acronyms, you need to use a glossary style that displays the symbol key.

```

6326 \newcommand*\SetDescriptionFootnoteAcronymStyle{%
6327   \renewcommand{\newacronym}[4][\]{%
6328     \ifx\@glsacronymlists\@empty
6329       \def\@glo@type{\acronymtype}%
6330       \setkeys{glossentry}{##1}%
6331       \DeclareAcronymList{\@glo@type}%
6332       \SetDescriptionFootnoteAcronymDisplayStyle{\@glo@type}%
6333     \fi
6334     \glskeylisttok{##1}%
6335     \glslabeltok{##2}%
6336     \glsshorttok{##3}%
6337     \gslongtok{##4}%
6338     \newacronymhook
6339     \DescriptionFootnoteNewAcronymDef
6340   }%

```

If footnote package option is specified, set the first use to append the long form (stored in symbol) as a footnote.

```

6341 \@for\@gls@type:=\@glsacronymlists\do{%
6342   \SetDescriptionFootnoteAcronymDisplayStyle{\@gls@type}%
6343 }%

```

Redefine `\acronymfont` if small caps required. The plural suffix is set in an upright font so that it remains in normal lower case, otherwise it looks as though it's part of the acronym.

```

6344 \ifglsacrsmallcaps
6345   \renewcommand*\acronymfont{[1]{\textsc{##1}}}%
6346   \renewcommand*\acrpluralsuffix{%
6347     \glstextup{\glspluralsuffix}}%
6348   \else
6349     \ifglsacrsmaller
6350       \renewcommand*\acronymfont{[1]{\textsmaller{##1}}}%
6351     \fi
6352   \fi

```

Check for package option clash

```

6353 \ifglsacrdua
6354   \PackageError{glossaries}{Option clash: ‘footnote’ and ‘dua’

```

```

6355     can't both be set}{}%
6356   \fi
6357 }%

```

AcronymDisplayStyle Sets the acronym display style for given glossary with description and dua combination.

```

6358 \newcommand*{\SetDescriptionDUAAcronymDisplayStyle}[1]{%
6359   \def\glentryfmt[#1]{\glsgenentryfmt}%
6360 }

```

ionDUANewAcronymDef

```

6361 \newcommand*{\DescriptionDUANewAcronymDef}{%
6362   \edef\@do@newglossaryentry{%
6363     \noexpand\newglossaryentry{\the\glslabeltok}%
6364     {%
6365       type=\acronymtype,%
6366       name={\the\glslongtok},%
6367       sort={\the\glslongtok},%
6368       text={\the\glslongtok},%
6369       first={\the\glslongtok},%
6370       plural={\noexpand\expandonce\noexpand\@glo@longpl},%
6371       firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6372       short={\the\glsshorttok},%
6373       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6374       long={\the\glslongtok},%
6375       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6376       symbol={\the\glsshorttok},%
6377       symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6378       \the\glskeylisttok
6379     }%
6380   }%
6381   \let\@org@gls@assign@firstpl\gls@assign@firstpl
6382   \let\@org@gls@assign@plural\gls@assign@plural
6383   \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6384   \def\gls@assign@firstpl##1##2{%
6385     \@@gls@expand@field{##1}{firstpl}{##2}%
6386   }%
6387   \def\gls@assign@plural##1##2{%
6388     \@@gls@expand@field{##1}{plural}{##2}%
6389   }%
6390   \def\gls@assign@symbolplural##1##2{%
6391     \@@gls@expand@field{##1}{symbolplural}{##2}%
6392   }%
6393   \@do@newglossaryentry
6394   \let\gls@assign@firstpl\@org@gls@assign@firstpl
6395   \let\gls@assign@plural\@org@gls@assign@plural
6396   \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6397 }

```

tionDUAAcronymStyle Description, don't use acronym and no footnote. Note that the short form is stored in the symbol key, so if the short form needs to be displayed in the glossary, use a style the displays the symbol.

```

6398 \newcommand*{\SetDescriptionDUAAcronymStyle}{%
6399   \ifglsmallcaps
6400     \PackageError{glossaries}{Option clash: 'smallcaps' and 'dua'
6401       can't both be set}{}%
6402   \else
6403     \ifglsmaller
6404       \PackageError{glossaries}{Option clash: 'smaller' and 'dua'
6405         can't both be set}{}%
6406     \fi
6407   \fi
6408   \renewcommand{\newacronym}[4][[]]{%
6409     \ifx\@glsacronymlists\@empty
6410       \def\@glo@type{\acronymtype}%
6411       \setkeys{glossentry}{##1}%
6412       \DeclareAcronymList{\@glo@type}%
6413       \SetDescriptionDUAAcronymDisplayStyle{\@glo@type}%
6414     \fi
6415     \glskeylisttok{##1}%
6416     \glslabeltok{##2}%
6417     \glsshorttok{##3}%
6418     \glslongtok{##4}%
6419     \newacronymhook
6420     \DescriptionDUANewAcronymDef
6421   }%

Set display.
6422   \@for\@gls@type:=\@glsacronymlists\do{%
6423     \SetDescriptionDUAAcronymDisplayStyle{\@gls@type}%
6424   }%
6425 }%

```

AcronymDisplayStyle Sets the acronym display style for given glossary using the description setting (but not footnote or dua).

```

6426 \newcommand*{\SetDescriptionAcronymDisplayStyle}[1]{%
6427   \defglentryfmt[#1]{%

6428     \ifdefempty\glscustomtext
6429     {%
6430       \ifglused{\glslabel}%
6431       {%

Move the inserted text outside of \acronymfont
6432         \let\gls@org@insert\glsinsert
6433         \let\glsinsert\@empty
6434         \acronymfont{\glsgenentryfmt}\gls@org@insert
6435       }%

```



```

6436      {%
6437      \glsgenentryfmt
6438      \ifglshassymbol{\glslabel}%
6439      {%
6440      \glsifplural
6441      {%
6442      \def\@glo@symbol{\glsentrysymbolplural{\glslabel}}%
6443      }%
6444      {%
6445      \def\@glo@symbol{\glsentrysymbol{\glslabel}}%
6446      }%
6447      \space(\protect\firstacronymfont
6448      {\glscapscase
6449      {\@glo@symbol}
6450      {\@glo@symbol}
6451      {\mfirstucMakeUppercase{\@glo@symbol}}})%
6452      }%
6453      }%
6454      }%
6455      }%
6456      {\glscustomtext\glsinsert}%
6457      }%
6458      }

```

ptionNewAcronymDef

```

6459 \newcommand*{\DescriptionNewAcronymDef}{%
6460 \edef\@do@newglossaryentry{%
6461 \noexpand\newglossaryentry{\the\glslabeltok}%
6462 {%
6463 type=\acronymtype,%
6464 name={\noexpand
6465 \acrnameformat{\the\glsshorttok}{\the\glslongtok}},%
6466 sort={\the\glsshorttok},%
6467 first={\the\glslongtok},%
6468 firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6469 text={\the\glsshorttok},%
6470 plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6471 short={\the\glsshorttok},%
6472 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6473 long={\the\glslongtok},%
6474 longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6475 symbol={\noexpand\@glo@text},%
6476 symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6477 \the\glskeylisttok}%
6478 }%
6479 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6480 \let\@org@gls@assign@plural\gls@assign@plural
6481 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6482 \def\gls@assign@firstpl##1##2{%

```

```

6483 \@@gls@expand@field{##1}{firstpl}{##2}%
6484 }%
6485 \def\gls@assign@plural##1##2{%
6486 \@@gls@expand@field{##1}{plural}{##2}%
6487 }%
6488 \def\gls@assign@symbolplural##1##2{%
6489 \@@gls@expand@field{##1}{symbolplural}{##2}%
6490 }%
6491 \do@newglossaryentry
6492 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6493 \let\gls@assign@plural\@org@gls@assign@plural
6494 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6495 }

```

DescriptionAcronymStyle Option description is used, but not dua or footnote. Store long form in first key and short form in text and symbol key. The name is stored using `\acrnameformat` to allow the user to override the way the name is displayed in the list of acronyms.

```

6496 \newcommand*{\SetDescriptionAcronymStyle}{%
6497 \renewcommand{\newacronym}[4][\]{%
6498 \ifx\@glsacronymlists\@empty
6499 \def\@glo@type{\acronymtype}%
6500 \setkeys{glossentry}{##1}%
6501 \DeclareAcronymList{\@glo@type}%
6502 \SetDescriptionAcronymDisplayStyle{\@glo@type}%
6503 \fi
6504 \glskeylisttok{##1}%
6505 \glslabeltok{##2}%
6506 \glsshorttok{##3}%
6507 \glslongtok{##4}%
6508 \newacronymhook
6509 \DescriptionNewAcronymDef
6510 }%

```

Set display.

```

6511 \@for\@gls@type:=\@glsacronymlists\do{%
6512 \SetDescriptionAcronymDisplayStyle{\@gls@type}%
6513 }%

```

Redefine `\acronymfont` if small caps required. The plural suffix is set in an upright font so that it remains in normal lower case, otherwise it looks as though it's part of the acronym.

```

6514 \ifglsacrsmallcaps
6515 \renewcommand{\acronymfont}[1]{\textsc{##1}}
6516 \renewcommand*{\acrpluralsuffix}{%
6517 \glstextup{\glspluralsuffix}}%
6518 \else
6519 \ifglsacrsmaller
6520 \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%

```

```

6521 \fi
6522 \fi
6523 }%

```

AcronymDisplayStyle Sets the acronym display style for given glossary with footnote setting (but not description or dua).

```

6524 \newcommand*{\SetFootnoteAcronymDisplayStyle}[1]{%
6525 \def\glsentryfmt[#1]{%

```

```

6526 \ifdefempty\glscustomtext
6527 {%

```

Move the inserted text outside of \acronymfont

```

6528 \let\gls@org@insert\glsinsert
6529 \let\glsinsert\@empty
6530 \ifglssused{\glslabel}%
6531 {%
6532 \acronymfont{\glsentryfmt}\gls@org@insert
6533 }%
6534 {%
6535 \firstacronymfont{\glsentryfmt}\gls@org@insert
6536 \ifglshaslong{\glslabel}%
6537 {%
6538 \expandafter\protect\expandafter\acrfootnote\expandafter
6539 {\@gls@link@opts}{\@gls@link@label}%
6540 {%
6541 \glsifplural
6542 {\glsentrylongpl{\glslabel}}%
6543 {\glsentrylong{\glslabel}}%
6544 }%
6545 }%
6546 {}%
6547 }%
6548 }%
6549 {\glscustomtext\glsinsert}%
6550 }%
6551 }

```

FootnoteNewAcronymDef

```

6552 \newcommand*{\FootnoteNewAcronymDef}{%
6553 \edef\@do@newglossaryentry{%
6554 \noexpand\newglossaryentry{\the\glslabeltok}%
6555 {%
6556 type=\acronymtype,%
6557 name={\noexpand\acronymfont{\the\glsshorttok}},%
6558 sort={\the\glsshorttok},%
6559 text={\the\glsshorttok},%
6560 plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6561 first={\the\glsshorttok},%

```

```

6562     firstplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6563     short={\the\glsshorttok},%
6564     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6565     long={\the\glslongtok},%
6566     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6567     description={\the\glslongtok},%
6568     descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6569     \the\glskeylisttok
6570 }%
6571 }%
6572 \let\@org@gls@assign@plural\gls@assign@plural
6573 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6574 \let\@org@gls@assign@descplural\gls@assign@descplural
6575 \def\gls@assign@firstpl##1##2{%
6576   \@@gls@expand@field{##1}{firstpl}{##2}%
6577 }%
6578 \def\gls@assign@plural##1##2{%
6579   \@@gls@expand@field{##1}{plural}{##2}%
6580 }%
6581 \def\gls@assign@descplural##1##2{%
6582   \@@gls@expand@field{##1}{descplural}{##2}%
6583 }%
6584 \do@newglossaryentry
6585 \let\gls@assign@plural\@org@gls@assign@plural
6586 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6587 \let\gls@assign@descplural\@org@gls@assign@descplural
6588 }

```

footnoteAcronymStyle If footnote package option is specified, set the first use to append the long form (stored in description) as a footnote. Use the description key to store the long form.

```

6589 \newcommand*\SetFootnoteAcronymStyle{%
6590   \renewcommand{\newacronym}[4][\]{%
6591     \ifx\@glsacronymlists\@empty
6592       \def\@glo@type{\acronymtype}%
6593       \setkeys{glossentry}{##1}%
6594       \DeclareAcronymList{\@glo@type}%
6595       \SetFootnoteAcronymDisplayStyle{\@glo@type}%
6596     \fi
6597     \glskeylisttok{##1}%
6598     \glslabeltok{##2}%
6599     \glsshorttok{##3}%
6600     \glslongtok{##4}%
6601     \newacronymhook
6602     \FootnoteNewAcronymDef
6603   }%

```

Set display

```

6604   \@for\@gls@type:=\@glsacronymlists\do{%

```

```

6605 \SetFootnoteAcronymDisplayStyle{\@gls@type}%
6606 }%

```

Redefine \acronymfont if small caps required. The plural suffix is set in an upright font so that it remains in normal lower case, otherwise it looks as though it's part of the acronym.

```

6607 \ifglsacrsmallcaps
6608 \renewcommand*{\acronymfont}[1]{\textsc{##1}}%
6609 \renewcommand*{\acrpluralsuffix}{%
6610 \glstextup{\glspluralsuffix}}%
6611 \else
6612 \ifglsacrsmaller
6613 \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%
6614 \fi
6615 \fi

```

Check for option clash

```

6616 \ifglsacrdua
6617 \PackageError{glossaries}{Option clash: ‘footnote’ and ‘dua’
6618 can’t both be set}{}%
6619 \fi
6620 }%

```

\glsdoparenifnotempty Do a space followed by the argument if the argument doesn't expand to empty or \relax. If argument isn't empty (or \relax), apply the macro to it given in the second argument.

```

6621 \DeclareRobustCommand*{\glsdoparenifnotempty}[2]{%
6622 \protected@edef\gls@tmp{#1}%
6623 \ifdefempty\gls@tmp
6624 {}%
6625 {%
6626 \ifx\gls@tmp\@gls@default@value
6627 \else
6628 \space (#2{#1})%
6629 \fi
6630 }%
6631 }

```

AcronymDisplayStyle Sets the acronym display style for given glossary where neither footnote nor description is required, but smallcaps or smaller specified.

```

6632 \newcommand*{\SetSmallAcronymDisplayStyle}[1]{%
6633 \defglsentryfmt[#1]{%
6634 \ifdefempty\glscustomtext
6635 {%

```

Move the inserted text outside of \acronymfont

```

6636 \let\gls@org@insert\glsinsert
6637 \let\glsinsert\@empty
6638 \ifglsused{\glslabel}%

```

```

6639      {%
6640      \acronymfont{\glsgenentryfmt}\gls@org@insert
6641      }%
6642      {%
6643      \glsgenentryfmt
6644      \ifglshassymbol{\glslabel}%
6645      {%
6646      \glsifplural
6647      {%
6648      \def\@glo@symbol{\glsentrysymbolplural{\glslabel}}%
6649      }%
6650      {%
6651      \def\@glo@symbol{\glsentrysymbol{\glslabel}}%
6652      }%
6653      \space
6654      (\glscapscase
6655      {\firstacronymfont{\@glo@symbol}}%
6656      {\firstacronymfont{\@glo@symbol}}%
6657      {\firstacronymfont{\mfirstucMakeUppercase{\@glo@symbol}}})%
6658      }%
6659      {}%
6660      }%
6661      }%
6662      {\glscustomtext\glsinsert}%
6663      }%
6664      }

```

\SmallNewAcronymDef

```

6665 \newcommand*{\SmallNewAcronymDef}{%
6666   \edef\@do@newglossaryentry{%
6667     \noexpand\newglossaryentry{\the\glslabeltok}%
6668     {%
6669       type=\acronymtype,%
6670       name={\noexpand\acronymfont{\the\glsshorttok}},%
6671       sort={\the\glsshorttok},%
6672       text={\the\glsshorttok},%

```

Default to the short plural.

```

6673     plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6674     first={\the\glslongtok},%

```

Default to the long plural.

```

6675     firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6676     short={\the\glsshorttok},%
6677     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6678     long={\the\glslongtok},%
6679     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6680     description={\noexpand\@glo@first},%
6681     descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6682     symbol={\the\glsshorttok},%

```

Default to the short plural.

```

6683     symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6684     \the\glskeylisttok
6685   }%
6686 }%
6687 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6688 \let\@org@gls@assign@plural\gls@assign@plural
6689 \let\@org@gls@assign@descplural\gls@assign@descplural
6690 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6691 \def\gls@assign@firstpl##1##2{%
6692   \@@gls@expand@field{##1}{firstpl}{##2}%
6693 }%
6694 \def\gls@assign@plural##1##2{%
6695   \@@gls@expand@field{##1}{plural}{##2}%
6696 }%
6697 \def\gls@assign@descplural##1##2{%
6698   \@@gls@expand@field{##1}{descplural}{##2}%
6699 }%
6700 \def\gls@assign@symbolplural##1##2{%
6701   \@@gls@expand@field{##1}{symbolplural}{##2}%
6702 }%
6703 \@do@newglossaryentry
6704 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6705 \let\gls@assign@plural\@org@gls@assign@plural
6706 \let\gls@assign@descplural\@org@gls@assign@descplural
6707 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6708 }

```

`\setSmallAcronymStyle` Neither footnote nor description required, but smallcaps or smaller specified.
Use the symbol key to store the short form and first to store the long form.

```

6709 \newcommand*{\SetSmallAcronymStyle}{%
6710   \renewcommand{\newacronym}[4][[]]{%
6711     \ifx\@glsacronymlists\@empty
6712       \def\@glo@type{\acronymtype}%
6713       \setkeys{glossentry}{##1}%
6714       \DeclareAcronymList{\@glo@type}%
6715       \SetSmallAcronymDisplayStyle{\@glo@type}%
6716     \fi
6717     \glskeylisttok{##1}%
6718     \glslabeltok{##2}%
6719     \glsshorttok{##3}%
6720     \glslongtok{##4}%
6721     \newacronymhook
6722     \SmallNewAcronymDef
6723   }%

```

Change the display since first only contains long form.

```

6724   \@for\@gls@type:=\@glsacronymlists\do{%
6725     \SetSmallAcronymDisplayStyle{\@gls@type}%

```

6726 }%

Redefine \acronymfont if small caps required. The plural suffix is set in an up-right font so that it remains in normal lower case, otherwise it looks as though it's part of the acronym.

```
6727 \ifglsmallcaps
6728   \renewcommand*{\acronymfont}[1]{\textsc{##1}}
6729   \renewcommand*{\acrpluralsuffix}{%
6730     \glstextup{\glspluralsuffix}}%
6731 \else
6732   \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}
6733 \fi
```

check for option clash

```
6734 \ifglsmallcaps
6735   \ifglsmallcaps
6736     \PackageError{glossaries}{Option clash: 'smallcaps' and 'dua'
6737       can't both be set}{}%
6738   \else
6739     \PackageError{glossaries}{Option clash: 'smaller' and 'dua'
6740       can't both be set}{}%
6741 \fi
6742 \fi
6743 }%
```

\SetDUADisplayStyle Sets the acronym display style for given glossary with dua setting.

```
6744 \newcommand*{\SetDUADisplayStyle}[1]{%
6745   \defglentryfmt[##1]{\glsgenentryfmt}%
6746 }
```

\DUANewAcronymDef

```
6747 \newcommand*{\DUANewAcronymDef}{%
6748   \edef\@do@newglossaryentry{%
6749     \noexpand\newglossaryentry{\the\glslabeltok}%
6750     {%
6751       type=\acronymtype,%
6752       name={\the\glsshorttok},%
6753       text={\the\glslongtok},%
6754       first={\the\glslongtok},%
6755       plural={\noexpand\expandonce\noexpand\@glo@longpl},%
6756       firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6757       short={\the\glsshorttok},%
6758       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6759       long={\the\glslongtok},%
6760       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6761       description={\the\glslongtok},%
6762       descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6763       symbol={\the\glsshorttok},%
6764       symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6765       \the\glskeylisttok
```



```

6766 }%
6767 }%
6768 \let\@org@gl@s@assign@firstpl\gl@s@assign@firstpl
6769 \let\@org@gl@s@assign@plural\gl@s@assign@plural
6770 \let\@org@gl@s@assign@symbolplural\gl@s@assign@symbolplural
6771 \let\@org@gl@s@assign@descplural\gl@s@assign@descplural
6772 \def\gl@s@assign@firstpl##1##2{%
6773   \@@gl@s@expand@field{##1}{firstpl}{##2}%
6774 }%
6775 \def\gl@s@assign@plural##1##2{%
6776   \@@gl@s@expand@field{##1}{plural}{##2}%
6777 }%
6778 \def\gl@s@assign@symbolplural##1##2{%
6779   \@@gl@s@expand@field{##1}{symbolplural}{##2}%
6780 }%
6781 \def\gl@s@assign@descplural##1##2{%
6782   \@@gl@s@expand@field{##1}{descplural}{##2}%
6783 }%
6784 \do@newglossaryentry
6785 \let\gl@s@assign@firstpl\@org@gl@s@assign@firstpl
6786 \let\gl@s@assign@plural\@org@gl@s@assign@plural
6787 \let\gl@s@assign@symbolplural\@org@gl@s@assign@symbolplural
6788 \let\gl@s@assign@descplural\@org@gl@s@assign@descplural
6789 }

```

\SetDUASyle Always expand acronyms.

```

6790 \newcommand*\SetDUASyle{%
6791   \renewcommand{\newacronym}[4][\]{%
6792     \ifx\@gl@s@acronymlists\@empty
6793       \def\@glo@type{\acronymtype}%
6794       \setkeys{glossentry}{##1}%
6795       \DeclareAcronymList{\@glo@type}%
6796       \SetDUADisplayStyle{\@glo@type}%
6797     \fi
6798     \gl@s@keylisttok{##1}%
6799     \gl@s@labeltok{##2}%
6800     \gl@s@shorttok{##3}%
6801     \gl@s@longtok{##4}%
6802     \newacronymhook
6803     \DUANewAcronymDef
6804   }%

```

Set the display

```

6805   \@for\@gl@s@type:=\@gl@s@acronymlists\do{%
6806     \SetDUADisplayStyle{\@gl@s@type}%
6807   }%
6808 }

```

\SetAcronymStyle

```

6809 \newcommand*{\SetAcronymStyle}{%
6810   \SetDefaultAcronymStyle
6811   \ifglssacrdescription
6812     \ifglssacrfootnote
6813       \SetDescriptionFootnoteAcronymStyle
6814     \else
6815       \ifglssacrdua
6816         \SetDescriptionDUAAcronymStyle
6817       \else
6818         \SetDescriptionAcronymStyle
6819     \fi
6820   \fi
6821 \else
6822   \ifglssacrfootnote
6823     \SetFootnoteAcronymStyle
6824   \else
6825     \ifthenelse{\boolean{glssacrsmallcaps}}\OR
6826       \boolean{glssacrsmaller}}{%
6827     {%
6828       \SetSmallAcronymStyle
6829     }%
6830     {%
6831       \ifglssacrdua
6832         \SetDUASstyle
6833       \fi
6834     }%
6835   \fi
6836 \fi
6837 }

```

Set the acronym style according to the package options

```
6838 \SetAcronymStyle
```

Allow user to define their own custom acronyms. (For compatibility with versions before v3.0, the short form is stored in the user1 key, the plural short form is stored in the user2 key, the long form is stored in the user3 key and the plural long form is stored in the user4 key.) Defaults to displaying only the acronym with the long form as the description.

CustomDisplayStyle Sets the acronym display style.

```

6839 \newcommand*{\SetCustomDisplayStyle}[1]{%
6840   \defglssentryfmt[#1]{\glsgenentryfmt}%
6841 }

```

CustomAcronymFields

```

6842 \newcommand*{\CustomAcronymFields}{%
6843   name={\the\glssshorttok},%
6844   description={\the\glslongtok},%
6845   first={\acrfullformat{\the\glslongtok}{\the\glssshorttok}},%

```

```

6846 firstplural={\acrfullformat
6847   {\noexpand\glsentrylongpl{\the\glslabeltok}}}%
6848   {\noexpand\glsentryshortpl{\the\glslabeltok}}},%

6849 text={\the\glsshorttok},%
6850 plural={\the\glsshorttok\noexpand\acrpluralsuffix}%
6851 }

```

CustomNewAcronymDef

```

6852 \newcommand*{\CustomNewAcronymDef}{%
6853   \protected@edef\@do@newglossaryentry{%
6854     \noexpand\newglossaryentry{\the\glslabeltok}%
6855     {%
6856       type=\acronymtype,%
6857       short={\the\glsshorttok},%
6858       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6859       long={\the\glslongtok},%
6860       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6861       user1={\the\glsshorttok},%
6862       user2={\the\glsshorttok\noexpand\acrpluralsuffix},%
6863       user3={\the\glslongtok},%
6864       user4={\the\glslongtok\noexpand\acrpluralsuffix},%
6865       \CustomAcronymFields,%
6866       \the\glskeylisttok
6867     }%
6868   }%
6869   \@do@newglossaryentry
6870 }

```

\SetCustomStyle

```

6871 \newcommand*{\SetCustomStyle}{%
6872   \renewcommand{\newacronym}[4][\]{%
6873     \ifx\@glsacronymlists\@empty
6874       \def\@glo@type{\acronymtype}%
6875       \setkeys{glossentry}{##1}%
6876       \DeclareAcronymList{\@glo@type}%
6877       \SetCustomDisplayStyle{\@glo@type}%
6878     \fi
6879     \glskeylisttok{##1}%
6880     \glslabeltok{##2}%
6881     \glsshorttok{##3}%
6882     \glslongtok{##4}%
6883     \newacronymhook
6884     \CustomNewAcronymDef
6885   }%

   Set the display
6886   \@for\@gls@type:=\@glsacronymlists\do{%
6887     \SetCustomDisplayStyle{\@gls@type}%
6888   }%

```

6889 }

1.18 Predefined Glossary Styles

The glossaries bundle comes with some predefined glossary styles. These need to be loaded now for the style option to use them.

First, the glossary hyper-navigation commands need to be loaded.

6890 \RequirePackage{glossary-hypernav}

The styles that use list-like environments. These are not loaded if the nolist option is used:

6891 \@gls@loadlist

The styles that use the longtable environment. These are not loaded if the no-long package option is used.

6892 \@gls@loadlong

The styles that use the supertabular environment. These are not loaded if the nosuper package option is used or if the package isn't installed.

6893 \@gls@loadsuper

The tree-like styles. These are not loaded if the notree package option is used.

6894 \@gls@loadtree

The default glossary style is set according to the style package option, but can be overridden by \glossarystyle. The required style must be defined at this point.

6895 \ifx\@glossary@default@style\relax

6896 \else

6897 \setglossarystyle{\@glossary@default@style}

6898 \fi

1.19 Debugging Commands

\showgloparent \showgloparent{<label>}

6899 \newcommand*{\showgloparent}[1]{%

6900 \expandafter\show\csname glo@\glsdetoklabel{#1}@parent\endcsname

6901 }

\showglolevel \showglolevel{<label>}

6902 \newcommand*{\showglolevel}[1]{%

6903 \expandafter\show\csname glo@\glsdetoklabel{#1}@level\endcsname

6904 }

`\showglotext` `\showglotext{<label>}`

```
6905 \newcommand*{\showglotext}[1]{%
6906   \expandafter\show\csname glo@glstetoklabel{#1}@text\endcsname
6907 }
```

`\showgloplural` `\showgloplural{<label>}`

```
6908 \newcommand*{\showgloplural}[1]{%
6909   \expandafter\show\csname glo@glstetoklabel{#1}@plural\endcsname
6910 }
```

`\showglofirst` `\showglofirst{<label>}`

```
6911 \newcommand*{\showglofirst}[1]{%
6912   \expandafter\show\csname glo@glstetoklabel{#1}@first\endcsname
6913 }
```

`\showglofirstpl` `\showglofirstpl{<label>}`

```
6914 \newcommand*{\showglofirstpl}[1]{%
6915   \expandafter\show\csname glo@glstetoklabel{#1}@firstpl\endcsname
6916 }
```

`\showglotype` `\showglotype{<label>}`

```
6917 \newcommand*{\showglotype}[1]{%
6918   \expandafter\show\csname glo@glstetoklabel{#1}@type\endcsname
6919 }
```

`\showglocounter` `\showglocounter{<label>}`

```
6920 \newcommand*{\showglocounter}[1]{%
6921   \expandafter\show\csname glo@glstetoklabel{#1}@counter\endcsname
6922 }
```

\showglouserii \showglouserii{\label{}}

```
6923 \newcommand*{\showglouserii}[1]{%
6924   \expandafter\show\csname glo@\glsdetoklabel{#1}@userii\endcsname
6925 }
```

\showglouseriii \showglouseriii{\label{}}

```
6926 \newcommand*{\showglouseriii}[1]{%
6927   \expandafter\show\csname glo@\glsdetoklabel{#1}@useriii\endcsname
6928 }
```

\showglouseriv \showglouseriv{\label{}}

```
6929 \newcommand*{\showglouseriv}[1]{%
6930   \expandafter\show\csname glo@\glsdetoklabel{#1}@useriv\endcsname
6931 }
```

\showglouserv \showglouserv{\label{}}

```
6932 \newcommand*{\showglouserv}[1]{%
6933   \expandafter\show\csname glo@\glsdetoklabel{#1}@userv\endcsname
6934 }
```

\showglouservi \showglouservi{\label{}}

```
6935 \newcommand*{\showglouservi}[1]{%
6936   \expandafter\show\csname glo@\glsdetoklabel{#1}@uservi\endcsname
6937 }
```

\showglouservi \showglouservi{\label{}}

```
6938 \newcommand*{\showglouservi}[1]{%
6939   \expandafter\show\csname glo@\glsdetoklabel{#1}@uservi\endcsname
6940 }
```

`\showglo`name `\showglo`name{<label>}

```
6941 \newcommand*{\showglo
```

```
name}[1]{%
```

```
6942   \expandafter\show\csname glo@\glsdetoklabel{#1}@name\endcsname
```

```
6943 }
```

`\showglo`desc `\showglo`desc{<label>}

```
6944 \newcommand*{\showglo
```

```
desc}[1]{%
```

```
6945   \expandafter\show\csname glo@\glsdetoklabel{#1}@desc\endcsname
```

```
6946 }
```

`\showglo`descplural `\showglo`descplural{<label>}

```
6947 \newcommand*{\showglo
```

```
descplural}[1]{%
```

```
6948   \expandafter\show\csname glo@\glsdetoklabel{#1}@descplural\endcsname
```

```
6949 }
```

`\showglo`sort `\showglo`sort{<label>}

```
6950 \newcommand*{\showglo
```

```
sort}[1]{%
```

```
6951   \expandafter\show\csname glo@\glsdetoklabel{#1}@sort\endcsname
```

```
6952 }
```

`\showglo`symbol `\showglo`symbol{<label>}

```
6953 \newcommand*{\showglo
```

```
symbol}[1]{%
```

```
6954   \expandafter\show\csname glo@\glsdetoklabel{#1}@symbol\endcsname
```

```
6955 }
```

`\showglo`symbolplural `\showglo`symbolplural{<label>}

```
6956 \newcommand*{\showglo
```

```
symbolplural}[1]{%
```

```
6957   \expandafter\show\csname glo@\glsdetoklabel{#1}@symbolplural\endcsname
```

```
6958 }
```

`\showgloshort` `\showgloshort{<label>}`

```
6959 \newcommand*{\showgloshort}[1]{%
6960   \expandafter\show\csname glo@\glsdetoklabel{#1}@short\endcsname
6961 }
```

`\showglolong` `\showglolong{<label>}`

```
6962 \newcommand*{\showglolong}[1]{%
6963   \expandafter\show\csname glo@\glsdetoklabel{#1}@long\endcsname
6964 }
```

`\showgloindex` `\showgloindex{<label>}`

```
6965 \newcommand*{\showgloindex}[1]{%
6966   \expandafter\show\csname glo@\glsdetoklabel{#1}@index\endcsname
6967 }
```

`\showgloflag` `\showgloflag{<label>}`

```
6968 \newcommand*{\showgloflag}[1]{%
6969   \expandafter\show\csname ifglo@\glsdetoklabel{#1}@flag\endcsname
6970 }
```

`\showgloloclist` `\showgloloclist{<label>}`

```
6971 \newcommand*{\showgloloclist}[1]{%
6972   \expandafter\show\csname glo@\glsdetoklabel{#1}@loclist\endcsname
6973 }
```

`\showacronymlists` `\showacronymlists`

Show list of glossaries that have been flagged as a list of acronyms.

```
6974 \newcommand*{\showacronymlists}{%
6975   \show\@glsacronymlists
6976 }
```


`\showglossaries` `\showglossaries`

Show list of defined glossaries.

```
6977 \newcommand*{\showglossaries}{%  
6978   \show\@glo@types  
6979 }
```

`\showglossaryin` `\showglossaryin{<glossary-label>}`

Show the ‘in’ extension for the given glossary.

```
6980 \newcommand*{\showglossaryin}[1]{%  
6981   \expandafter\show\csname @glotype@#1@in\endcsname  
6982 }
```

`\showglossaryout` `\showglossaryout{<glossary-label>}`

Show the ‘out’ extension for the given glossary.

```
6983 \newcommand*{\showglossaryout}[1]{%  
6984   \expandafter\show\csname @glotype@#1@out\endcsname  
6985 }
```

`\showglossarytitle` `\showglossarytitle{<glossary-label>}`

Show the title for the given glossary.

```
6986 \newcommand*{\showglossarytitle}[1]{%  
6987   \expandafter\show\csname @glotype@#1@title\endcsname  
6988 }
```

`\showglossarycounter` `\showglossarycounter{<glossary-label>}`

Show the counter for the given glossary.

```
6989 \newcommand*{\showglossarycounter}[1]{%  
6990   \expandafter\show\csname @glotype@#1@counter\endcsname  
6991 }
```

`\showglossaryentries` `\showglossaryentries{<glossary-label>}`

Show the list of entry labels for the given glossary.

```
6992 \newcommand*{\showglossaryentries}[1]{%  
6993   \expandafter\show\csname glolist@#1\endcsname  
6994 }
```

1.20 Compatibility with version 2.07 and below

In order to fix some bugs in v3.0, it was necessary to change the way information is written to the `glo` file, which also meant a change in the format of the Xindy style file. The compatibility option is meant for documents that use a customised Xindy style file with `\noist`. With the compatibility option, hopefully xindy will still be able to process the old document, but the bugs will remain. The issues in versions 2.07 and below:

- With xindy, the counter used by the entry was hard-coded into the Xindy style file. This meant that you couldn't use the counter to swap counters.
- With both xindy and makeindex, if used with hyperref and `\theH{counter}` was different to `\thecounter`, the link in the location number would be undefined.

```
6995 \csname ifglscompatible-2.07\endcsname
6996 \RequirePackage{glossaries-compatible-207}
6997 \fi
```

2 Prefix Support (glossaries-prefix Code)

This package provides a means of adding prefixes to your glossary entries. For example, you may want to use “a `\gls{<label>}`” on first use but use “an `\gls{<label>}`” on subsequent use.

```
6998 \NeedsTeXFormat{LaTeX2e}
6999 \ProvidesPackage{glossaries-prefix}[2014/07/30 v4.08 (NLCT)]

Pass all options to glossaries:
7000 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{glossaries}}

Process options:
7001 \ProcessOptions

Load glossaries:
7002 \RequirePackage{glossaries}

Add the new keys:
7003 \define@key{glossentry}{prefixfirst}{\def\@glo@entryprefixfirst{#1}}%
7004 \define@key{glossentry}{prefixfirstplural}{\def\@glo@entryprefixfirstplural{#1}}%
7005 \define@key{glossentry}{prefix}{\def\@glo@entryprefix{#1}}%
7006 \define@key{glossentry}{prefixplural}{\def\@glo@entryprefixplural{#1}}%

Add them to \@gls@keymap:
7007 \appto\@gls@keymap{,%
7008   {prefixfirst}{prefixfirst},%
7009   {prefixfirstplural}{prefixfirstplural},%
7010   {prefix}{prefix},%
7011   {prefixplural}{prefixplural}}%
7012 }
```

Set the default values:

```

7013 \appto\@newglossaryentryprehook{%
7014   \def\@glo@entryprefix{}%
7015   \def\@glo@entryprefixplural{}%
7016   \let\@glo@entryprefixfirst\@gls@default@value
7017   \let\@glo@entryprefixfirstplural\@gls@default@value
7018 }

```

Set the assignment code:

```

7019 \appto\@newglossaryentryposthook{%
7020   \gls@assign@field{\@glo@label}{prefix}{\@glo@entryprefix}%
7021   \gls@assign@field{\@glo@label}{prefixplural}{\@glo@entryprefixplural}%

```

If prefixfirst has not been supplied, make it the same as prefix.

```

7022   \expandafter\gls@assign@field\expandafter
7023     {\csname glo@\@glo@label @prefix\endcsname}{\@glo@label}{prefixfirst}%
7024     {\@glo@entryprefixfirst}%

```

If prefixfirstplural has not been supplied, make it the same as prefixplural.

```

7025   \expandafter\gls@assign@field\expandafter
7026     {\csname glo@\@glo@label @prefixplural\endcsname}{\@glo@label}%
7027     {prefixfirstplural}{\@glo@entryprefixfirstplural}%
7028 }

```

Define commands to access these fields:

glsentryprefixfirst

```

7029 \newcommand*\{glsentryprefixfirst}[1]{\csuse{glo@#1@prefixfirst}}

```

ryprefixfirstplural

```

7030 \newcommand*\{glsentryprefixfirstplural}[1]{\csuse{glo@#1@prefixfirstplural}}

```

\glsentryprefix

```

7031 \newcommand*\{glsentryprefix}[1]{\csuse{glo@#1@prefix}}

```

lsentryprefixplural

```

7032 \newcommand*\{glsentryprefixplural}[1]{\csuse{glo@#1@prefixplural}}

```

Now for the initial upper case variants:

Glsentryprefixfirst

```

7033 \newrobustcmd*\{Glsentryprefixfirst}[1]{%
7034   \protected@edef\@glo@text{\csname glo@#1@prefixfirst\endcsname}%
7035   \xmakefirstuc\@glo@text
7036 }

```

ryprefixfirstplural

```

7037 \newrobustcmd*\{Glsentryprefixfirstplural}[1]{%
7038   \protected@edef\@glo@text{\csname glo@#1@prefixfirstplural\endcsname}%
7039   \xmakefirstuc\@glo@text
7040 }

```

\Glsentryprefix

```
7041 \newrobustcmd*{\Glsentryprefix}[1]{%
7042   \protected@edef\@glo@text{\csname glo@#1@prefix\endcsname}%
7043   \xmakefirstuc\@glo@text
7044 }
```

\Glsentryprefixplural

```
7045 \newrobustcmd*{\Glsentryprefixplural}[1]{%
7046   \protected@edef\@glo@text{\csname glo@#1@prefixplural\endcsname}%
7047   \xmakefirstuc\@glo@text
7048 }
```

Define commands to determine if the prefix keys have been set:

\ifglshasprefix

```
7049 \newcommand*{\ifglshasprefix}[3]{%
7050   \ifcempty{glo@#1@prefix}%
7051   {#3}%
7052   {#2}%
7053 }
```

\ifglshasprefixplural

```
7054 \newcommand*{\ifglshasprefixplural}[3]{%
7055   \ifcempty{glo@#1@prefixplural}%
7056   {#3}%
7057   {#2}%
7058 }
```

\ifglshasprefixfirst

```
7059 \newcommand*{\ifglshasprefixfirst}[3]{%
7060   \ifcempty{glo@#1@prefixfirst}%
7061   {#3}%
7062   {#2}%
7063 }
```

\ifglshasprefixfirstplural

```
7064 \newcommand*{\ifglshasprefixfirstplural}[3]{%
7065   \ifcempty{glo@#1@prefixfirstplural}%
7066   {#3}%
7067   {#2}%
7068 }
```

Define commands that insert the prefix before commands like \gls:

\pgls

```
7069 \newrobustcmd{\pgls}{\@gls@hyp@opt\@pgls}
```

\@pgls Unstarred version.

```
7070 \newcommand*{\@pgls}[2] [] {%
7071   \new@ifnextchar [%
7072     {\@pgls@{#1}{#2}}%
7073     {\@pgls@{#1}{#2} []}%
7074 }
```

\@pgls@ Read in the final optional argument:

```
7075 \def\@pgls@#1#2[#3] {%
7076   \glsdoifexists{#2}%
7077   {%
7078     \ifglsused{#2}%
7079     {%
7080       \glsentryprefix{#2}%
7081     }%
7082     {%
7083       \glsentryprefixfirst{#2}%
7084     }%
7085     \@gls@{#1}{#2}[#3]%
7086   }%
7087 }
```

Similarly for the plural version:

\pglsp1

```
7088 \newrobustcmd{\pglsp1}{\@gls@hyp@opt\@pglsp1}
```

\@pglsp1 Unstarred version.

```
7089 \newcommand*{\@pglsp1}[2] [] {%
7090   \new@ifnextchar [%
7091     {\@pglsp1@{#1}{#2}}%
7092     {\@pglsp1@{#1}{#2} []}%
7093 }
```

\@pglsp1@ Read in the final optional argument:

```
7094 \def\@pglsp1@#1#2[#3] {%
7095   \glsdoifexists{#2}%
7096   {%
7097     \ifglsused{#2}%
7098     {%
7099       \glsentryprefixplural{#2}%
7100     }%
7101     {%
7102       \glsentryprefixfirstplural{#2}%
7103     }%
7104     \@glspl@{#1}{#2}[#3]%
7105   }%
7106 }
```

Now for the first letter upper case versions:

\Pgl's

```
7107 \newrobustcmd{\Pgl's}{\@gls@hyp@opt\@Pgl's}
```

\@Pgl's Unstarred version.

```
7108 \newcommand*{\@Pgl's}[2] [] {%
7109   \new@ifnextchar[%
7110     {\@Pgl's@{#1}{#2}}}%
7111   {\@Pgl's@{#1}{#2} []}%
7112 }
```

\@Pgl's@ Read in the final optional argument:

```
7113 \def\@Pgl's@#1#2[#3] {%
7114   \glsdoifexists{#2}%
7115   {%
7116     \ifglsused{#2}%
7117     {%
7118       \ifgls prefix{#2}%
7119       {%
7120         \Glsentryprefix{#2}%
7121         \@gls@{#1}{#2}[#3]%
7122       }%
7123       {\@Gls@{#1}{#2}[#3]}%
7124     }%
7125     {%
7126       \ifgls prefixfirst{#2}%
7127       {%
7128         \Glsentryprefixfirst{#2}%
7129         \@gls@{#1}{#2}[#3]%
7130       }%
7131       {\@Gls@{#1}{#2}[#3]}%
7132     }%
7133   }%
7134 }
```

Similarly for the plural version:

\Pgl'spl

```
7135 \newrobustcmd{\Pgl'spl}{\@gls@hyp@opt\@Pgl'spl}
```

\@Pgl'spl Unstarred version.

```
7136 \newcommand*{\@Pgl'spl}[2] [] {%
7137   \new@ifnextchar[%
7138     {\@Pgl'spl@{#1}{#2}}}%
7139   {\@Pgl'spl@{#1}{#2} []}%
7140 }
```

\@Pglsp1@ Read in the final optional argument:

```
7141 \def\@Pglsp1@#1#2[#3]{%
7142   \glsdoifexists{#2}%
7143   {%
7144     \ifglused{#2}%
7145     {%
7146       \ifglshasprefixplural{#2}%
7147       {%
7148         \Glsentryprefixplural{#2}%
7149         \@glsp1@{#1}{#2}[#3]%
7150       }%
7151       {\@Glspl@{#1}{#2}[#3]}%
7152     }%
7153     {%
7154       \ifglshasprefixfirstplural{#2}%
7155       {%
7156         \Glsentryprefixfirstplural{#2}%
7157         \@glsp1@{#1}{#2}[#3]%
7158       }%
7159       {\@Glspl@{#1}{#2}[#3]}%
7160     }%
7161   }%
7162 }
```

Finally the all upper case versions:

\PGLS

```
7163 \newrobustcmd{\PGLS}{\@gls@hyp@opt\@PGLS}
```

\@PGLS Unstarred version.

```
7164 \newcommand*{\@PGLS}[2][ ]{%
7165   \new@ifnextchar[%
7166   {\@PGLS@{#1}{#2}}%
7167   {\@PGLS@{#1}{#2}[ ]}%
7168 }
```

\@PGLS@ Read in the final optional argument:

```
7169 \def\@PGLS@#1#2[#3]{%
7170   \glsdoifexists{#2}%
7171   {%
7172     \ifglused{#2}%
7173     {%
7174       \mfirstucMakeUppercase{\glsentryprefix{#2}}%
7175     }%
7176     {%
7177       \mfirstucMakeUppercase{\glsentryprefixfirst{#2}}%
7178     }%
7179     \@GLS@{#1}{#2}[#3]%
7180   }
```

```

7180 }%
7181 }

```

Plural version:

`\PGLSp1`

```

7182 \newrobustcmd{\PGLSp1}{\@gls@hyp@opt\PGLSp1}

```

`\@PGLSp1` Unstarred version.

```

7183 \newcommand*{\@PGLSp1}[2][{}]{%
7184   \new@ifnextchar[%
7185     {\@PGLSp1@{#1}{#2}}%
7186     {\@PGLSp1@{#1}{#2}[]}%
7187 }

```

`\@PGLSp1@` Read in the final optional argument:

```

7188 \def\@PGLSp1@#1#2[#3]{%
7189   \glsdoifexists{#2}%
7190   {%
7191     \ifglsused{#2}%
7192     {%
7193       \mfirstucMakeUppercase{\glsentryprefixplural{#2}}%
7194     }%
7195     {%
7196       \mfirstucMakeUppercase{\glsentryprefixfirstplural{#2}}%
7197     }%
7198     \@GLSp1@{#1}{#2}[#3]%
7199   }%
7200 }

```

3 Mfirstuc Documented Code

```

7201 \NeedsTeXFormat{LaTeX2e}
7202 \ProvidesPackage{mfirstuc}[2014/07/30 v1.09 (NLCT)]

```

Requires etoolbox:

```

7203 \RequirePackage{etoolbox}

```

`\makefirstuc` Syntax:

`\makefirstuc{<text>}`

Makes the first letter uppercase, but will skip initial control sequences if they are followed by a group and make the first thing in the group uppercase, unless the group is empty. Thus `\makefirstuc{abc}` will produce: `Abc`, `\makefirstuc{\ae bc}` will produce: `Æbc`, but `\makefirstuc{\emph{abc}}` will produce `Abc`. This is required by `\Gls` and `\Glspl`.

```

7204 \newif\if@glscs

```



```

7205 \newtoks\@glsmfirst
7206 \newtoks\@glsmrest
7207 \newrobustcmd*{\makefirstuc}[1]{%
7208   \def\gls@argi{#1}%
7209   \ifx\gls@argi\@empty

       If the argument is empty, do nothing.
7210   \else

7211     \def\@gls@tmp{\ #1}%
7212     \@onelevel@sanitize\@gls@tmp
7213     \expandafter\@gls@checkcs\@gls@tmp\relax\relax
7214     \if@glscs
7215       \@gls@getbody #1{}\@nil
7216       \ifx\@gls@rest\@empty
7217         \glsmakefirstuc{#1}%
7218       \else
7219         \expandafter\@gls@split\@gls@rest\@nil
7220         \ifx\@gls@first\@empty
7221           \glsmakefirstuc{#1}%
7222         \else
7223           \expandafter\@glsmfirst\expandafter{\@gls@first}%
7224           \expandafter\@glsmrest\expandafter{\@gls@rest}%
7225           \edef\@gls@domfirstuc{\noexpand\@gls@body
7226             {\noexpand\glsmakefirstuc\the\@glsmfirst}%
7227             \the\@glsmrest}%
7228           \@gls@domfirstuc
7229         \fi
7230       \fi
7231     \else
7232       \glsmakefirstuc{#1}%
7233     \fi
7234   \fi
7235 }

       Put first argument in \@gls@first and second argument in \@gls@rest:
7236 \def\@gls@split#1#2\@nil{%
7237   \def\@gls@first{#1}\def\@gls@rest{#2}%
7238 }

7239 \def\@gls@checkcs#1 #2#3\relax{%
7240   \def\@gls@argi{#1}\def\@gls@argii{#2}%
7241   \ifx\@gls@argi\@gls@argii
7242     \@glscstrue
7243   \else
7244     \@glscsfalse
7245   \fi
7246 }

```

\@gls@makefirstuc Make first thing upper case:

```

7247 \def\@gls@makefirstuc#1{\mfirstucMakeUppercase #1}

```

irstucMakeUppercase Allow user to replace \MakeUppercase with another case changing command.

```
7248 \newcommand*{\mfirstucMakeUppercase}{\MakeUppercase}
```

\glsmakefirstuc Provide a user command to make it easier to customise.

```
7249 \newcommand*{\glsmakefirstuc}[1]{\@gls@makefirstuc{#1}}
```

Get the first grouped argument and stores in \@gls@body.

```
7250 \def\@gls@getbody#1#\def\@gls@body{#1}\@gls@gobbletonil}
```

Scoup up everything to \@nil and store in \@gls@rest:

```
7251 \def\@gls@gobbletonil#1\@nil{\def\@gls@rest{#1}}
```

\xmakefirstuc Expand argument once before applying \makefirstuc (added v1.01).

```
7252 \newcommand*{\xmakefirstuc}[1]{%
```

```
7253 \expandafter\makefirstuc\expandafter{#1}}
```

\capitalisewords Capitalise each word in the argument. Words are considered to be separated by plain spaces (i.e. non-breakable spaces won't be considered a word break).

```
7254 \newrobustcmd*{\capitalisewords}[1]{%
```

```
7255 \def\gls@add@space{}%
```

```
7256 \let\@mfu@domakefirstuc\makefirstuc
```

```
7257 \let\@mfu@checkword\@gobble
```

```
7258 \mfu@capitalisewords#1 \@nil\mfu@endcap
```

```
7259 }
```

```
7260 \def\mfu@capitalisewords#1 #2\mfu@endcap{%
```

```
7261 \def\mfu@cap@first{#1}%
```

```
7262 \def\mfu@cap@second{#2}%
```

```
7263 \gls@add@space
```

```
7264 \@mfu@checkword{#1}%
```

```
7265 \@mfu@domakefirstuc{#1}%
```

```
7266 \def\gls@add@space{ }%
```

```
7267 \ifx\mfu@cap@second\@nnil
```

```
7268 \let\next\mfu@cap\mfu@noop
```

```
7269 \else
```

```
7270 \let\next\mfu@cap\mfu@capitalisewords
```

```
7271 \let\@mfu@checkword\mfu@checkword
```

```
7272 \fi
```

```
7273 \next\mfu@cap#2\mfu@endcap
```

```
7274 }
```

```
7275 \def\mfu@noop#1\mfu@endcap{}
```

\mfu@checkword Check if word should be capitalised.

```
7276 \newcommand*{\mfu@checkword}[1]{%
```

```
7277 \ifinlist{#1}{\@mfu@nocaplist}%
```

```
7278 {%
```

```
7279 \let\@mfu@domakefirstuc\@firstofone
```

```
7280 }%
```

```
7281 {%
```

```

7282 \let\@mfu@domakefirstuc\makefirstuc
7283 }%
7284 }

```

\@mfu@nocaplist List of words that shouldn't be capitalised.

```
7285 \newcommand*\@mfu@nocaplist{}
```

\MFUnocap Provide the user with a means to add a word to the list.

```
7286 \newcommand*\MFUnocap[1]{\listadd{\@mfu@nocaplist}{#1}}
```

\gMFUnocap Global version.

```
7287 \newcommand*\gMFUnocap[1]{\listgadd{\@mfu@nocaplist}{#1}}
```

\MFUclear Clear the list

```
7288 \newcommand*\MFUclear{\renewcommand*\@mfu@nocaplist{}}
```

\xcapitalisewords Short-cut command:

```

7289 \newcommand*\xcapitalisewords[1]{%
7290 \expandafter\capitalisewords\expandafter{#1}%
7291 }

```

4 Mfirstuc-english Documented Code

```

7292 \NeedsTeXFormat{LaTeX2e}
7293 \ProvidesPackage{mfirstuc-english}[2014/07/30 v1.0 (NLCT)]

```

Load mfirstuc if not already loaded:

```
7294 \RequirePackage{mfirstuc}
```

Add no-cap words. (List isn't a complete list.)

```

7295 \MFUnocap{a}
7296 \MFUnocap{an}
7297 \MFUnocap{and}
7298 \MFUnocap{but}
7299 \MFUnocap{for}
7300 \MFUnocap{in}
7301 \MFUnocap{of}
7302 \MFUnocap{or}
7303 \MFUnocap{no}
7304 \MFUnocap{nor}
7305 \MFUnocap{so}
7306 \MFUnocap{some}
7307 \MFUnocap{the}
7308 \MFUnocap{with}
7309 \MFUnocap{yet}

```

5 Glossary Styles

5.1 Glossary hyper-navigation definitions (glossary-hypernav package)

Package Definition:

```
7310 \ProvidesPackage{glossary-hypernav}[2013/11/14 v4.0 (NLCT)]
```

The commands defined in this package are provided to help navigate around the groups within a glossary (see [subsection 1.15.](#)) `\printglossary` (and `\printglossaries`) set `\@glo@type` to the label of the current glossary. This is used to create a unique hypertarget in the event of multiple glossaries.

`\glsnavhyperlink[⟨type⟩]{⟨label⟩}{⟨text⟩}`

This command makes `⟨text⟩` a hyperlink to the glossary group whose label is given by `⟨label⟩` for the glossary given by `⟨type⟩`.

`\glsnavhyperlink`

```
7311 \newcommand*{\glsnavhyperlink}[3][\@glo@type]{%
7312   \edef\gls@grplabel{#2}\protected@edef\gls@grptitle{#3}%
7313   \@glslink{glsn:#1@#2}{#3}}
```

`\glsnavhypertarget[⟨type⟩]{⟨label⟩}{⟨text⟩}`

This command makes `⟨text⟩` a hypertarget for the glossary group whose label is given by `⟨label⟩` in the glossary given by `⟨type⟩`. If `⟨type⟩` is omitted, `\@glo@type` is used which is set by `\printglossary` to the current glossary label.

`\glsnavhypertarget`

```
7314 \newcommand*{\glsnavhypertarget}[3][\@glo@type]{%
  Add this group to the aux file for re-run check.
7315   \protected@write\@auxout{}{\string\@gls@hypergroup{#1}{#2}}%
  Add the target.
7316   \@glstarget{glsn:#1@#2}{#3}%
  Check list of know groups to determine if a re-run is required.
7317   \expandafter\let
7318     \expandafter\@gls@list\csname @gls@hypergroup@#1\endcsname
  Iterate through list and terminate loop if this group is found.
7319   \@for\@gls@elem:=\@gls@list\do{%
7320     \ifthenelse{\equal{\@gls@elem}{#2}}{\@endfortrue}{}%
  Check if list terminated prematurely.
7321   \if@endfor
7322   \else
```

This group was not included in the list, so issue a warning.

```

7323 \GlossariesWarningNoLine{Navigation panel
7324   for glossary type ‘#1’^^Jmissing group ‘#2’}%
7325 \gdef\gls@hypergroupprerun{%
7326   \GlossariesWarningNoLine{Navigation panel
7327   has changed. Rerun LaTeX}}%
7328 \fi
7329 }

```

`\gls@hypergroupprerun` Give a warning at the end if re-run required

```

7330 \let\gls@hypergroupprerun\relax
7331 \AtEndDocument{\gls@hypergroupprerun}

```

`\@gls@hypergroup` This adds to (or creates) the command `\@gls@hypergrouplist@<glossary type>` which lists all groups for a given glossary, so that the navigation bar only contains those groups that are present. However it requires at least 2 runs to ensure the information is up-to-date.

```

7332 \newcommand*{\@gls@hypergroup}[2]{%
7333 \@ifundefined{\@gls@hypergrouplist@#1}{%
7334   \expandafter\xdef\csname @gls@hypergrouplist@#1\endcsname{#2}%
7335 }{%
7336   \expandafter\let\expandafter\@gls@tmp
7337   \csname @gls@hypergrouplist@#1\endcsname
7338   \expandafter\xdef\csname @gls@hypergrouplist@#1\endcsname{%
7339     \@gls@tmp,#2}%
7340 }%
7341 }

```

The `\glsnavigation` command displays a simple glossary group navigation. The symbol and number elements are defined separately, so that they can be suppressed if need be. Note that this command will produce a link to all 28 groups, but some groups may not be defined if there are groups that do not contain any terms, in which case you will get an undefined hyperlink warning. Now for the whole navigation bit:

`\glsnavigation`

```

7342 \newcommand*{\glsnavigation}{%
7343 \def\@gls@between{}%
7344 \@ifundefined{\@gls@hypergrouplist@\@glo@type}{%
7345   \def\@gls@list{}%
7346 }{%
7347   \expandafter\let\expandafter\@gls@list
7348   \csname @gls@hypergrouplist@\@glo@type\endcsname
7349 }%
7350 \@for\@gls@tmp:=\@gls@list\do{%
7351   \@gls@between
7352   \@gls@getgrouptitle{\@gls@tmp}{\@gls@grptitle}%

```

```

7353 \glsnavhyperlink{\@gls@tmp}{\@gls@grptitle}%
7354 \let\@gls@between\glshypernavsep%
7355 }%
7356 }

```

`\glshypernavsep` Separator for the hyper navigation bar.

```

7357 \newcommand*{\glshypernavsep}{\space\textbar\space}

```

The `\glssymbolnav` produces a simple navigation set of links for just the symbol and number groups. This used to be used at the start of `\glsnavigation`. This command is no longer needed.

`\glssymbolnav`

```

7358 \newcommand*{\glssymbolnav}{%
7359 \glsnavhyperlink{glssymbols}{\glsgetgrouptitle{glssymbols}}%
7360 \glshypernavsep
7361 \glsnavhyperlink{glsnumbers}{\glsgetgrouptitle{glsnumbers}}%
7362 \glshypernavsep
7363 }

```

5.2 In-line Style (glossary-inline.sty)

This defines an in-line style where the entries are comma-separated with just the name and description displayed.

```

7364 \ProvidesPackage{glossary-inline}[2013/11/14 v4.0 (NLCT)]

```

`inline` Define the inline style.

```

7365 \newglossarystyle{inline}{%

```

Start of glossary sets up first empty separator between entries. (This is then changed by `\glossentry`)

```

7366 \renewenvironment{theglossary}%
7367 {%
7368 \def\gls@inlinesep{}%
7369 \def\gls@inlinesubsep{}%
7370 \def\gls@inlinepostchild{}%
7371 }%
7372 {\glspostinline}%

```

No header:

```

7373 \renewcommand*{\glossaryheader}{}%

```

No group headings (if heading is required, add `\glsinlinedopostchild` to start definition in case heading follows a child entry):

```

7374 \renewcommand*{\glsgroupheading}[1]{}%

```

Just display separator followed by name and description:

```

7375 \renewcommand{\glossentry}[2]{%
7376 \glsinlinedopostchild
7377 \gls@inlinesep

```

```

7378 \glsentryitem{##1}%
7379 \glsinlinenameformat{##1}{%
7380 \glossentryname{##1}%
7381 }%
7382 \ifglshasdescsuppressed{##1}%
7383 {%
7384 \glsinlineemptydescformat
7385 {%
7386 \glossentrysymbol{##1}%
7387 }%
7388 {%
7389 ##2%
7390 }%
7391 }%
7392 {%
7393 \ifglshasdesc{##1}%
7394 {\glsinlinedescformat{\glossentrydesc{##1}}{\glossentrysymbol{##1}}{##2}}%
7395 {\glsinlineemptydescformat{\glossentrysymbol{##1}}{##2}}%
7396 }%
7397 \ifglshaschildren{##1}%
7398 {%
7399 \glsresetsubentrycounter
7400 \glsinlineparentchildseparator
7401 \def\gls@inlinesubsep{}%
7402 \def\gls@inlinepostchild{\glsinlinepostchild}%
7403 }%
7404 {}%
7405 \def\gls@inlinesep{\glsinlineseparator}%
7406 }%

```

Sub-entries display description:

```

7407 \renewcommand{\subglossentry}[3]{%
7408 \gls@inlinesubsep%
7409 \glsinlinesubnameformat{##2}{%
7410 \glossentryname{##2}}%
7411 \glssubentryitem{##2}%
7412 \glsinlinesubdescformat{\glossentrydesc{##2}}{\glossentrysymbol{##2}}{##3}%
7413 \def\gls@inlinesubsep{\glsinlinesubseparator}%
7414 }%

```

Nothing special between groups:

```

7415 \renewcommand*{\glsgroupskip}{}%
7416 }

```

`\glsinlinedopostchild`

```

7417 \newcommand*{\glsinlinedopostchild}{%
7418 \gls@inlinepostchild
7419 \def\gls@inlinepostchild{}%
7420 }

```

`\glsinlineseparator` Separator to use between entries.
7421 `\newcommand*{\glsinlineseparator}{;\space}`

`\glsinlinesubseparator` Separator to use between sub-entries.
7422 `\newcommand*{\glsinlinesubseparator}{,\space}`

`\glsinlineparentchildseparator` Separator to use between parent and children.
7423 `\newcommand*{\glsinlineparentchildseparator}{:\space}`

`\glsinlinepostchild` Hook to use between child and next entry
7424 `\newcommand*{\glsinlinepostchild}{}`

`\glspostinline` Terminator for inline glossary.
7425 `\newcommand*{\glspostinline}{\glspostdescription\space}`

`\glsinlinenameformat` Formats the name of the entry (first argument label, second argument name):
7426 `\newcommand*{\glsinlinenameformat}[2]{\glstarget{#1}{#2}}`

`\glsinlinedescformat` Formats the entry's description, symbol and location list:
7427 `\newcommand*{\glsinlinedescformat}[3]{\space#1}`

`\glsinlineemptydescformat` Formats the entry's symbol and location list when the description is empty:
7428 `\newcommand*{\glsinlineemptydescformat}[2]{}`

`\glsinlinesubnameformat` Formats the name of the subentry (first argument label, second argument name):
7429 `\newcommand*{\glsinlinesubnameformat}[2]{\glstarget{#1}{}}`

`\glsinlinesubdescformat` Formats the subentry's description, symbol and location list:
7430 `\newcommand*{\glsinlinesubdescformat}[3]{#1}`

5.3 List Style (glossary-list.sty)

The style file defines glossary styles that use the description environment. Note that since the entry name is placed in the optional argument to the `\item` command, it will appear in a bold font by default.

7431 `\ProvidesPackage{glossary-list}[2013/11/14 v4.0 (NLCT)]`

`list` The list glossary style uses the description environment. The group separator `\glsgroupskip` is redefined as `\indexspace` which produces a gap between groups. The glossary heading and the group headings do nothing. Sub-entries immediately follow the main entry without the sub-entry name. This style does not use the entry's symbol. This is used as the default style for the glossaries package.

7432 `\newglossarystyle{list}{%`

Use description environment:

```
7433 \renewenvironment{theglossary}%  
7434   {\begin{description}}{\end{description}}%
```

No header at the start of the environment:

```
7435 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7436 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries start a new item in the list:

```
7437 \renewcommand*{\glossentry}[2]{%  
7438   \item[\glsentryitem{##1}]%  
7439     \glstarget{##1}{\glossentryname{##1}}]  
7440     \glossentrydesc{##1}\glspostdescription\space ##2}%
```

Sub-entries continue on the same line:

```
7441 \renewcommand*{\subglossentry}[3]{%  
7442   \glssubentryitem{##2}%  
7443   \glstarget{##2}{\strut}%  
7444   \glossentrydesc{##2}\glspostdescription\space ##3.}%  
7445% \end{macrocode}  
7446% Add vertical space between groups:  
7447%\changes{3.03}{2012/09/21}{added check for glsnogroupskip}  
7448% \begin{macrocode}  
7449 \renewcommand*{\glsgroupskip}{\ifglsgroupskip\else\indexspace\fi}%  
7450 }
```

listgroup The listgroup style is like the list style, but the glossary groups have headings.

```
7451 \newglossarystyle{listgroup}{%
```

Base it on the list style:

```
7452 \setglossarystyle{list}%
```

Each group has a heading:

```
7453 \renewcommand*{\glsgroupheading}[1]{\item[\glsgetgrouptitle{##1}]}}
```

listhypergroup The listhypergroup style is like the listgroup style, but has a set of links to the groups at the start of the glossary.

```
7454 \newglossarystyle{listhypergroup}{%
```

Base it on the list style:

```
7455 \setglossarystyle{list}%
```

Add navigation links at the start of the environment:

```
7456 \renewcommand*{\glossaryheader}{%  
7457   \item[\glsnavigation]}%
```

Each group has a heading with a hypertarget:

```
7458 \renewcommand*{\glsgroupheading}[1]{%  
7459   \item[\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}]}}
```

altlist The altlist glossary style is like the list style, but places the description on a new line. Sub-entries follow in separate paragraphs without the sub-entry name. This style does not use the entry's symbol.

```
7460 \newglossarystyle{altlist}{%
```

Base it on the list style:

```
7461 \setglossarystyle{list}%
```

Main (level 0) entries start a new item in the list with a line break after the entry name:

```
7462 \renewcommand*{\glossentry}[2]{%
7463   \item[\glsentryitem{##1}]%
7464   \glstarget{##1}{\glossentryname{##1}}}%
```

Version 3.04 changed \newline to the following paragraph break stuff (thanks to Daniel Gebhardt for supplying the fix) to prevent a page break occurring at this point.

```
7465   \mbox{}\par\nobreak\@afterheading
7466   \glossentrydesc{##1}\glspostdescription\space ##2}%
```

Sub-entries start a new paragraph:

```
7467 \renewcommand{\subglossentry}[3]{%
7468   \par
7469   \glssubentryitem{##2}%
7470   \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space ##3}%
7471 }
```

altlistgroup The altlistgroup glossary style is like the altlist style, but the glossary groups have headings.

```
7472 \newglossarystyle{altlistgroup}{%
```

Base it on the altlist style:

```
7473 \setglossarystyle{altlist}%
```

Each group has a heading:

```
7474 \renewcommand*{\glsgroupheading}[1]{\item[\glsgrouptitle{##1}]}
```

altlisthypergroup The altlisthypergroup glossary style is like the altlistgroup style, but has a set of links to the groups at the start of the glossary.

```
7475 \newglossarystyle{altlisthypergroup}{%
```

Base it on the altlist style:

```
7476 \setglossarystyle{altlist}%
```

Add navigation links at the start of the environment:

```
7477 \renewcommand*{\glossaryheader}{%
7478   \item[\glsnavigation]}%
```

Each group has a heading with a hypertarget:

```
7479 \renewcommand*{\glsgroupheading}[1]{%
7480   \item[\glsnavhypertarget{##1}{\glsgrouptitle{##1}}]}
```

`listdotted` The `listdotted` glossary style was supplied by Axel Menzel. I've modified it slightly so that the distance from the start of the name to the end of the dotted line is specified by `\glslistdottedwidth`. Note that this style ignores the page numbers as well as the symbol. Sub-entries are displayed in the same way as top-level entries.

```
7481 \newglossarystyle{listdotted}{%
```

Base it on the list style:

```
7482 \setglossarystyle{list}{%
```

Each main (level 0) entry starts a new item:

```
7483 \renewcommand*{\glossentry}[2]{%
7484 \item[]\makebox[\glslistdottedwidth][l]{%
7485 \glentryitem{##1}%
7486 \glstarget{##1}{\glossentryname{##1}}%
7487 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}\glossentrydesc{##1}}%
```

Sub entries have the same format as main entries:

```
7488 \renewcommand*{\subglossentry}[3]{%
7489 \item[]\makebox[\glslistdottedwidth][l]{%
7490 \glssubentryitem{##2}%
7491 \glstarget{##2}{\glossentryname{##2}}%
7492 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}\glossentrydesc{##2}}%
7493 }
```

`\glslistdottedwidth`

```
7494 \newlength\glslistdottedwidth
7495 \setlength{\glslistdottedwidth}{.5\hsize}
```

`sublistdotted` This style is similar to the `glostylelistdotted` style, except that the main entries just have the name displayed.

```
7496 \newglossarystyle{sublistdotted}{%
```

Base it on the `listdotted` style:

```
7497 \setglossarystyle{listdotted}{%
```

Main (level 0) entries just display the name:

```
7498 \renewcommand*{\glossentry}[2]{%
7499 \item[\glentryitem{##1}\glstarget{##1}{\glossentryname{##1}}]}%
7500 }
```

5.4 Glossary Styles using `longtable` (the `glossary-long` package)

The glossary styles defined in the package used the `longtable` environment in the glossary.

```
7501 \ProvidesPackage{glossary-long}[2013/11/14 v4.0 (NLCT)]
```

Requires the package:

```
7502 \RequirePackage{longtable}
```

`\glsdescwidth` This is a length that governs the width of the description column. (There's a chance that the user may specify `nolong` and then load later, in which case `\glsdescwidth` may have already been defined by . The same goes for `\glspagelistwidth`.)

```
7503 \ifundefined{glsdescwidth}{%
7504   \newlength{glsdescwidth
7505   \setlength{glsdescwidth}{0.6\hsize}
7506 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column.

```
7507 \ifundefined{glspagelistwidth}{%
7508   \newlength{glspagelistwidth
7509   \setlength{glspagelistwidth}{0.1\hsize}
7510 }{}
```

`long` The long glossary style command which uses the `longtable` environment:

```
7511 \newglossarystyle{long}{%
```

Use `longtable` with two columns:

```
7512   \renewenvironment{theglossary}%
7513     {\begin{longtable}{lp{glsdescwidth}}}%
7514     {\end{longtable}}%
```

Do nothing at the start of the environment:

```
7515   \renewcommand*{\glossaryheader}{}%
```

No heading between groups:

```
7516   \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries displayed in a row:

```
7517   \renewcommand{\glossentry}[2]{%
7518     \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7519     \glossentrydesc{##1}\glspostdescription\space ##2\tabularnewline
7520   }%
```

Sub entries displayed on the following row without the name:

```
7521   \renewcommand{\subglossentry}[3]{%
7522     &
7523     \glssubentryitem{##2}%
7524     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space
7525     ##3\tabularnewline
7526   }%
```

Blank row between groups:

```
7527   \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else &
7528   \tabularnewline\fi}%
7529 }
```

`longborder` The `longborder` style is like the above, but with horizontal and vertical lines:

```
7530 \newglossarystyle{longborder}{%
```

Base it on the `glostylelong` style:

```
7531 \setglossarystyle{long}%
```

Use `longtable` with two columns with vertical lines between each column:

```
7532 \renewenvironment{theglossary}{%  
7533 \begin{longtable}{|l|p{\glstdescwidth}|}{\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
7534 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%  
7535 }
```

`longheader` The `longheader` style is like the `long` style but with a header:

```
7536 \newglossarystyle{longheader}{%
```

Base it on the `glostylelong` style:

```
7537 \setglossarystyle{long}%
```

Set the table's header:

```
7538 \renewcommand*{\glossaryheader}{%  
7539 \bfseries \entryname & \bfseries \descriptionname\tabularnewline\endhead}%  
7540 }
```

`longheaderborder` The `longheaderborder` style is like the `long` style but with a header and border:

```
7541 \newglossarystyle{longheaderborder}{%
```

Base it on the `glostylelongborder` style:

```
7542 \setglossarystyle{longborder}%
```

Set the table's header and add horizontal line to table's foot:

```
7543 \renewcommand*{\glossaryheader}{%  
7544 \hline\bfseries \entryname & \bfseries  
7545 \descriptionname\tabularnewline\hline  
7546 \endhead  
7547 \hline\endfoot}%  
7548 }
```

`long3col` The `long3col` style is like `long` but with 3 columns

```
7549 \newglossarystyle{long3col}{%
```

Use a `longtable` with 3 columns:

```
7550 \renewenvironment{theglossary}%  
7551 {\begin{longtable}{lp{\glstdescwidth}p{\glspagelistwidth}}}%  
7552 {\end{longtable}}%
```

No table header:

```
7553 \renewcommand*{\glossaryheader}{}%
```

No headings between groups:

```
7554 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```

7555 \renewcommand{\glossentry}[2]{%
7556     \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7557     \glossentrydesc{##1} & ##2\tabularnewline
7558 }%
```

Sub-entries on a separate row (no name, description in second column, page list in third column):

```

7559 \renewcommand{\subglossentry}[3]{%
7560     &
7561     \glssubentryitem{##2}%
7562     \glstarget{##2}{\strut}\glossentrydesc{##2} &
7563     ##3\tabularnewline
7564 }%
```

Blank row between groups:

```

7565 \renewcommand*{\glsgroupskip}{%
7566     \ifglsgroupskip\else & &\tabularnewline\fi}%
7567 }
```

long3colborder The long3colborder style is like the long3col style but with a border:

```

7568 \newglossarystyle{long3colborder}{%
```

Base it on the glostylelong3col style:

```

7569 \setglossarystyle{long3col}%
```

Use a longtable with 3 columns with vertical lines around them:

```

7570 \renewenvironment{theglossary}%
7571     {\begin{longtable}{|l|p{\glsgdescwidth}|p{\glspagelistwidth}|}%
7572     {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```

7573 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
7574 }
```

long3colheader The long3colheader style is like long3col but with a header row:

```

7575 \newglossarystyle{long3colheader}{%
```

Base it on the glostylelong3col style:

```

7576 \setglossarystyle{long3col}%
```

Set the table's header:

```

7577 \renewcommand*{\glossaryheader}{%
7578     \bfseries\entryname&\bfseries\descriptionname&
7579     \bfseries\pagelistname\tabularnewline\endhead}%
7580 }
```

long3colheaderborder The long3colheaderborder style is like the above but with a border

```

7581 \newglossarystyle{long3colheaderborder}{%
```

Base it on the `glostylelong3colborder` style:

```
7582 \setglossarystyle{long3colborder}{%
```

Set the table's header and add horizontal line at table's foot:

```
7583 \renewcommand*{\glossaryheader}{{%  
7584 \hline  
7585 \bfseries\entryname&\bfseries\descriptionname&  
7586 \bfseries\pagelistname\tabularnewline\hline\endhead  
7587 \hline\endfoot}%  
7588 }
```

`long4col` The `long4col` style has four columns where the third column contains the value of the associated symbol key.

```
7589 \newglossarystyle{long4col}{%
```

Use a `longtable` with 4 columns:

```
7590 \renewenvironment{theglossary}{%  
7591 {\begin{longtable}{llll}}%  
7592 {\end{longtable}}%
```

No table header:

```
7593 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7594 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a single row (name in first column, description in second column, symbol in third column, page list in last column):

```
7595 \renewcommand{\glossentry}[2]{%  
7596 \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &  
7597 \glossentrydesc{##1} &  
7598 \glossentrysymbol{##1} &  
7599 ##2\tabularnewline  
7600 }%
```

Sub entries on a single row with no name (description in second column, symbol in third column, page list in last column):

```
7601 \renewcommand{\subglossentry}[3]{%  
7602 &  
7603 \glssubentryitem{##2}%  
7604 \glstarget{##2}{\strut}\glossentrydesc{##2} &  
7605 \glossentrysymbol{##2} & ##3\tabularnewline  
7606 }%
```

Blank row between groups:

```
7607 \renewcommand*{\glsgroupskip}{%  
7608 \ifglsgnোগroupskip\else & & \tabularnewline\fi}%  
7609 }
```

`long4colheader` The `long4colheader` style is like `long4col` but with a header row.

```
7610 \newglossarystyle{long4colheader}{%
```

Base it on the `glostylelong4col` style:

```
7611 \setglossarystyle{long4col}%
```

Table has a header:

```
7612 \renewcommand*{\glossaryheader}{%  
7613   \bfseries\entryname&\bfseries\descriptionname&  
7614   \bfseries \symbolname&  
7615   \bfseries\pagelistname\tabularnewline\endhead}%  
7616 }
```

`long4colborder` The `long4colborder` style is like `long4col` but with a border.

```
7617 \newglossarystyle{long4colborder}{%
```

Base it on the `glostylelong4col` style:

```
7618 \setglossarystyle{long4col}%
```

Use a longtable with 4 columns surrounded by vertical lines:

```
7619 \renewenvironment{theglossary}%  
7620   {\begin{longtable}{|l|l|l|l|}}%  
7621   {\end{longtable}}%
```

Add horizontal lines to the head and foot of the table:

```
7622 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%  
7623 }
```

`long4colheaderborder` The `long4colheaderborder` style is like the above but with a border.

```
7624 \newglossarystyle{long4colheaderborder}{%
```

Base it on the `glostylelong4col` style:

```
7625 \setglossarystyle{long4col}%
```

Use a longtable with 4 columns surrounded by vertical lines:

```
7626 \renewenvironment{theglossary}%  
7627   {\begin{longtable}{|l|l|l|l|}}%  
7628   {\end{longtable}}%
```

Add table header and horizontal line at the table's foot:

```
7629 \renewcommand*{\glossaryheader}{%  
7630   \hline\bfseries\entryname&\bfseries\descriptionname&  
7631   \bfseries \symbolname&  
7632   \bfseries\pagelistname\tabularnewline\hline\endhead  
7633   \hline\endfoot}%  
7634 }
```

`altlong4col` The `altlong4col` style is like the `long4col` style but can have multiline descriptions and page lists.

```
7635 \newglossarystyle{altlong4col}{%
```

Base it on the `glostylelong4col` style:

```
7636 \setglossarystyle{long4col}%
```


Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
7637 \renewenvironment{theglossary}%
7638   {\begin{longtable}{lp{\glsdescwidth}lp{\glspagelistwidth}}}%
7639   {\end{longtable}}%
7640 }
```

altlong4colheader The altlong4colheader style is like altlong4col but with a header row.

```
7641 \newglossarystyle{altlong4colheader}{%
```

Base it on the glostylelong4colheader style:

```
7642 \setglossarystyle{long4colheader}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
7643 \renewenvironment{theglossary}%
7644   {\begin{longtable}{lp{\glsdescwidth}lp{\glspagelistwidth}}}%
7645   {\end{longtable}}%
7646 }
```

altlong4colborder The altlong4colborder style is like altlong4col but with a border.

```
7647 \newglossarystyle{altlong4colborder}{%
```

Base it on the glostylelong4colborder style:

```
7648 \setglossarystyle{long4colborder}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
7649 \renewenvironment{theglossary}%
7650   {\begin{longtable}{|lp{\glsdescwidth}|lp{\glspagelistwidth}|}%
7651   {\end{longtable}}%
7652 }
```

ong4colheaderborder The altlong4colheaderborder style is like the above but with a header as well as a border.

```
7653 \newglossarystyle{altlong4colheaderborder}{%
```

Base it on the glostylelong4colheaderborder style:

```
7654 \setglossarystyle{long4colheaderborder}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
7655 \renewenvironment{theglossary}%
7656   {\begin{longtable}{|lp{\glsdescwidth}|lp{\glspagelistwidth}|}%
7657   {\end{longtable}}%
7658 }
```

5.5 Glossary Styles using longtable (the glossary-longragged package)

The glossary styles defined in the package used the longtable environment in the glossary and use ragged right formatting for the multiline columns.

7659 \ProvidesPackage{glossary-longragged}[2014/07/30 v4.08 (NLCT)]

Requires the package:

7660 \RequirePackage{array}

Requires the package:

7661 \RequirePackage{longtable}

`\glsdescwidth` This is a length that governs the width of the description column. This may have already been defined.

7662 \@ifundefined{glsdescwidth}{%

7663 \newlength{glsdescwidth}

7664 \setlength{glsdescwidth}{0.6\hsize}

7665 }{}

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

7666 \@ifundefined{glspagelistwidth}{%

7667 \newlength{glspagelistwidth}

7668 \setlength{glspagelistwidth}{0.1\hsize}

7669 }{}

`longragged` The longragged glossary style is like the long but uses ragged right formatting for the description column.

7670 \newglossarystyle{longragged}{%

Use longtable with two columns:

7671 \renewenvironment{theglossary}{%

7672 {\begin{longtable}{l>{\raggedright}p{glsdescwidth}}}%

7673 {\end{longtable}}}%

Do nothing at the start of the environment:

7674 \renewcommand*{\glossaryheader}{}%

No heading between groups:

7675 \renewcommand*{\glsgroupheading}[1]{}%

Main (level 0) entries displayed in a row:

7676 \renewcommand{\glossentry}[2]{%

7677 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &

7678 \glossentrydesc{##1}\glspostdescription\space ##2%

7679 \tabularnewline

7680 }%

Sub entries displayed on the following row without the name:

```

7681 \renewcommand{\subglossentry}[3]{%
7682     &
7683     \glssubentryitem{##2}%
7684     \glstarget{##2}{\strut}\glossentrydesc{##2}%
7685     \glspostdescription\space ##3%
7686     \tabularnewline
7687 }%
```

Blank row between groups:

```

7688 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & \tabularnewline\fi}%
7689 }
```

longraggedborder The longraggedborder style is like the above, but with horizontal and vertical lines:

```

7690 \newglossarystyle{longraggedborder}{%
```

Base it on the glostylelongragged style:

```

7691 \setglossarystyle{longragged}%
```

Use longtable with two columns with vertical lines between each column:

```

7692 \renewenvironment{theglossary}{%
7693     \begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|}%
7694     {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```

7695 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
7696 }
```

longraggedheader The longraggedheader style is like the longragged style but with a header:

```

7697 \newglossarystyle{longraggedheader}{%
```

Base it on the glostylelongragged style:

```

7698 \setglossarystyle{longragged}%
```

Set the table's header:

```

7699 \renewcommand*{\glossaryheader}{%
7700     \bfseries \entryname & \bfseries \descriptionname
7701     \tabularnewline\endhead}%
7702 }
```

longraggedheaderborder The longraggedheaderborder style is like the longragged style but with a header and border:

```

7703 \newglossarystyle{longraggedheaderborder}{%
```

Base it on the glostylelongraggedborder style:

```

7704 \setglossarystyle{longraggedborder}%
```

Set the table's header and add horizontal line to table's foot:

```

7705 \renewcommand*{\glossaryheader}{%
7706     \hline\bfseries \entryname & \bfseries \descriptionname
```

```

7707 \tabularnewline\hline
7708 \endhead
7709 \hline\endfoot}%
7710 }

```

`longragged3col` The `longragged3col` style is like `longragged` but with 3 columns

```

7711 \newglossarystyle{longragged3col}{%

```

Use a longtable with 3 columns:

```

7712 \renewenvironment{theglossary}%
7713 {\begin{longtable}{l>{\raggedright}p{\glsgdescwidth}%
7714 >{\raggedright}p{\glspagelistwidth}}}%
7715 {\end{longtable}}%

```

No table header:

```

7716 \renewcommand*{\glossaryheader}{}%

```

No headings between groups:

```

7717 \renewcommand*{\glsgroupheading}[1]{}%

```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```

7718 \renewcommand{\glossentry}[2]{%
7719 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7720 \glossentrydesc{##1} & ##2\tabularnewline
7721 }%

```

Sub-entries on a separate row (no name, description in second column, page list in third column):

```

7722 \renewcommand{\subglossentry}[3]{%
7723 &
7724 \glssubentryitem{##2}%
7725 \glstarget{##2}{\strut}\glossentrydesc{##2} &
7726 ##3\tabularnewline
7727 }%

```

Blank row between groups:

```

7728 \renewcommand*{\glsgroupskip}{%
7729 \ifglsgnoglobskip\else & &\tabularnewline\fi}%
7730 }

```

`longragged3colborder` The `longragged3colborder` style is like the `longragged3col` style but with a border:

```

7731 \newglossarystyle{longragged3colborder}{%

```

Base it on the `glostylelongragged3col` style:

```

7732 \setglossarystyle{longragged3col}%

```

Use a longtable with 3 columns with vertical lines around them:

```

7733 \renewenvironment{theglossary}%
7734 {\begin{longtable}{|l|>{\raggedright}p{\glsgdescwidth}|}%
7735 >{\raggedright}p{\glspagelistwidth}|}%
7736 {\end{longtable}}%

```

Place horizontal lines at the head and foot of the table:

```
7737 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
7738 }
```

`longragged3colheader` The `longragged3colheader` style is like `longragged3col` but with a header row:

```
7739 \newglossarystyle{longragged3colheader}{%
```

Base it on the `glostylelongragged3col` style:

```
7740 \setglossarystyle{longragged3col}%
```

Set the table's header:

```
7741 \renewcommand*{\glossaryheader}{%
7742 \bfseries\entryname&\bfseries\descriptionname&
7743 \bfseries\pagelistname\tabularnewline\endhead}%
7744 }
```

`longragged3colheaderborder` The `longragged3colheaderborder` style is like the above but with a border

```
7745 \newglossarystyle{longragged3colheaderborder}{%
```

Base it on the `glostylelongragged3colborder` style:

```
7746 \setglossarystyle{longragged3colborder}%
```

Set the table's header and add horizontal line at table's foot:

```
7747 \renewcommand*{\glossaryheader}{%
7748 \hline
7749 \bfseries\entryname&\bfseries\descriptionname&
7750 \bfseries\pagelistname\tabularnewline\hline\endhead
7751 \hline\endfoot}%
7752 }
```

`altlongragged4col` The `altlongragged4col` style is like the `altlong4col` style defined in the package, except that ragged right formatting is used for the description and page list columns.

```
7753 \newglossarystyle{altlongragged4col}{%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
7754 \renewenvironment{theglossary}%
7755 {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%
7756 >{\raggedright}p{\glspagelistwidth}}}%
7757 {\end{longtable}}%
```

No table header:

```
7758 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7759 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a single row (name in first column, description in second column, symbol in third column, page list in last column):

```
7760 \renewcommand{\glossentry}[2]{}%
```

```

7761 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7762 \glossentrydesc{##1} & \glossentrysymbol{##1} &
7763 ##2\tabularnewline
7764 }%

```

Sub entries on a single row with no name (description in second column, symbol in third column, page list in last column):

```

7765 \renewcommand{\subglossentry}[3]{%
7766   &
7767   \glssubentryitem{##2}%
7768   \glstarget{##2}{\strut}\glossentrydesc{##2} &
7769   \glossentrysymbol{##2} & ##3\tabularnewline
7770 }%

```

Blank row between groups:

```

7771 \renewcommand*\glsgroupskip{%
7772   \ifglisnogroupskip\else & & \tabularnewline\fi}%
7773 }

```

longragged4colheader The `altlongragged4colheader` style is like `altlongragged4col` but with a header row.

```

7774 \newglossarystyle{altlongragged4colheader}{%

```

Base it on the `glostylealtlongragged4col` style:

```

7775 \setglossarystyle{altlongragged4col}%

```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```

7776 \renewenvironment{theglossary}%
7777   {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%
7778     >{\raggedright}p{\glspagelistwidth}}}%
7779   {\end{longtable}}%

```

Table has a header:

```

7780 \renewcommand*\glossaryheader{%
7781   \bfseries\entryname&\bfseries\descriptionname&
7782   \bfseries \symbolname&
7783   \bfseries\pagelistname\tabularnewline\endhead}%
7784 }

```

longragged4colborder The `altlongragged4colborder` style is like `altlongragged4col` but with a border.

```

7785 \newglossarystyle{altlongragged4colborder}{%

```

Base it on the `glostylealtlongragged4col` style:

```

7786 \setglossarystyle{altlongragged4col}%

```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```

7787 \renewenvironment{theglossary}%
7788   {\begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|l|%
7789     >{\raggedright}p{\glspagelistwidth}|}%
7790   {\end{longtable}}%

```

Add horizontal lines to the head and foot of the table:

```
7791 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
7792 }
```

`ged4colheaderborder` The `altlongragged4colheaderborder` style is like the above but with a header as well as a border.

```
7793 \newglossarystyle{altlongragged4colheaderborder}{%
```

Base it on the `glostylealtlongragged4col` style:

```
7794 \setglossarystyle{altlongragged4col}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
7795 \renewenvironment{theglossary}%
7796 {\begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|l|}%
7797 >{\raggedright}p{\glspagelistwidth}|}%
7798 {\end{longtable}}%
```

Add table header and horizontal line at the table's foot:

```
7799 \renewcommand*{\glossaryheader}{%
7800 \hline\bfseries\entryname&\bfseries\descriptionname&
7801 \bfseries \symbolname&
7802 \bfseries\pagelistname\tabularnewline\hline\endhead
7803 \hline\endfoot}%
7804 }
```

5.6 Glossary Styles using multicol (glossary-mcols.sty)

The style file defines glossary styles that use the `multicol` package. These use the tree-like glossary styles in a `multicol` environment.

```
7805 \ProvidesPackage{glossary-mcols}[2013/11/14 v4.0 (NLCT)]
```

Required packages:

```
7806 \RequirePackage{multicol}
7807 \RequirePackage{glossary-tree}
```

`\glscols` Define macro in which to store the number of columns. (Defaults to 2.)

```
7808 \newcommand*{\glscols}{2}
```

`mcolindex` Multi-column index style. Same as the index, but puts the glossary in multiple columns. (Ideally the glossary title should go in the optional argument of `multicols`, but the title isn't part of the glossary style.)

```
7809 \newglossarystyle{mcolindex}{%
7810 \setglossarystyle{index}%
7811 \renewenvironment{theglossary}%
7812 {%
```

```

7813 \begin{multicols}{\glsmcols}
7814 \setlength{\parindent}{0pt}%
7815 \setlength{\parskip}{0pt plus 0.3pt}%
7816 \let\item\@idxitem}%
7817 {\end{multicols}}%
7818 }

```

mcolindexgroup As mcolindex but has headings:

```

7819 \newglossarystyle{mcolindexgroup}{%
7820 \setglossarystyle{mcolindex}%
7821 \renewcommand*{\glsgroupheading}[1]{%
7822 \item\textbf{\glsgroupheading{##1}}\indexspace}%
7823 }

```

mcolindexhypergroup The mcolindexhypergroup style is like the mcolindexgroup style but has hyper navigation.

```

7824 \newglossarystyle{mcolindexhypergroup}{%
    Base it on the glostylemcolindex style:
7825 \setglossarystyle{mcolindex}%
    Put navigation links to the groups at the start of the glossary:
7826 \renewcommand*{\glossaryheader}{%
7827 \item\textbf{\glshypernavigation}\indexspace}%
    Add a heading for each group (with a target). The group's title is in bold followed
    by a vertical gap.
7828 \renewcommand*{\glsgroupheading}[1]{%
7829 \item\textbf{\glshypernavhypertarget{##1}}{\glsgroupheading{##1}}}%
7830 \indexspace}%
7831 }

```

mcoltree Multi-column index style. Same as the tree, but puts the glossary in multiple columns.

```

7832 \newglossarystyle{mcoltree}{%
7833 \setglossarystyle{tree}%
7834 \renewenvironment{theglossary}%
7835 {%
7836 \begin{multicols}{\glsmcols}
7837 \setlength{\parindent}{0pt}%
7838 \setlength{\parskip}{0pt plus 0.3pt}%
7839 }%
7840 {\end{multicols}}%
7841 }

```

mcoltreegroup Like the mcoltree style but the glossary groups have headings.

```

7842 \newglossarystyle{mcoltreegroup}{%
    Base it on the glostylemcoltree style:
7843 \setglossarystyle{mcoltree}%

```


Each group has a heading (in bold) followed by a vertical gap):

```
7844 \renewcommand{\glsgroupheading}[1]{\par
7845 \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
7846 }
```

mcoltreehypergroup The mcoltreehypergroup style is like the treegroup style, but has a set of links to the groups at the start of the glossary.

```
7847 \newglossarystyle{mcoltreehypergroup}{%
```

Base it on the glostylemcoltree style:

```
7848 \setglossarystyle{mcoltree}%
```

Put navigation links to the groups at the start of the theglossary environment:

```
7849 \renewcommand*{\glossaryheader}{%
7850 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
7851 \renewcommand*{\glsgroupheading}[1]{%
7852 \par\noindent
7853 \textbf{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}\par
7854 \indexspace}%
7855 }
```

mcoltreenoname Multi-column index style. Same as the treenoname, but puts the glossary in multiple columns.

```
7856 \newglossarystyle{mcoltreenoname}{%
7857 \setglossarystyle{treenoname}%
7858 \renewenvironment{theglossary}%
7859 {%
7860 \begin{multicols}{\glsmcols}
7861 \setlength{\parindent}{0pt}%
7862 \setlength{\parskip}{0pt plus 0.3pt}%
7863 }%
7864 {\end{multicols}}}%
7865 }
```

mcoltreenonamegroup Like the mcoltreenoname style but the glossary groups have headings.

```
7866 \newglossarystyle{mcoltreenonamegroup}{%
```

Base it on the glostylemcoltreenoname style:

```
7867 \setglossarystyle{mcoltreenoname}%
```

Give each group a heading:

```
7868 \renewcommand{\glsgroupheading}[1]{\par
7869 \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
7870 }
```

treenonamehypergroup The mcoltreenonamehypergroup style is like the mcoltreenonamegroup style, but has a set of links to the groups at the start of the glossary.

```
7871 \newglossarystyle{mcoltreenonamehypergroup}{%
```

Base it on the `glostylemcoltreenoname` style:

```
7872 \setglossarystyle{mcoltreenoname}%
```

Put navigation links to the groups at the start of the `theglossary` environment:

```
7873 \renewcommand*{\glossaryheader}{%
```

```
7874 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
7875 \renewcommand*{\glsgroupheading}[1]{%
```

```
7876 \par\noindent
```

```
7877 \textbf{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}\par
```

```
7878 \indexspace}%
```

```
7879 }
```

`mcolalmtree` Multi-column index style. Same as the `almtree`, but puts the glossary in multiple columns.

```
7880 \newglossarystyle{mcolalmtree}{%
```

```
7881 \setglossarystyle{almtree}%
```

```
7882 \renewenvironment{theglossary}%
```

```
7883 {%
```

```
7884 \begin{multicols}{\glsmcols}
```

```
7885 \def\@gls@prevlevel{-1}%
```

```
7886 \mbox{}\par
```

```
7887 }%
```

```
7888 {\par\end{multicols}}%
```

```
7889 }
```

`mcolalmtreegroup` Like the `mcolalmtree` style but the glossary groups have headings.

```
7890 \newglossarystyle{mcolalmtreegroup}{%
```

Base it on the `glostylemcolalmtree` style:

```
7891 \setglossarystyle{mcolalmtree}%
```

Give each group a heading.

```
7892 \renewcommand{\glsgroupheading}[1]{\par
```

```
7893 \def\@gls@prevlevel{-1}%
```

```
7894 \hangindent0pt\relax
```

```
7895 \parindent0pt\relax
```

```
7896 \textbf{\glsgrouptitle{##1}}\par\indexspace}%
```

```
7897 }
```

`olalmtreehypergroup` The `mcolalmtreehypergroup` style is like the `mcolalmtreegroup` style, but has a set of links to the groups at the start of the glossary.

```
7898 \newglossarystyle{mcolalmtreehypergroup}{%
```

Base it on the `glostylemcolalmtree` style:

```
7899 \setglossarystyle{mcolalmtree}%
```

Put the navigation links in the header

```
7900 \renewcommand*{\glossaryheader}{%
7901   \par
7902   \def\@gls@prevlevel{-1}%
7903   \hangindent0pt\relax
7904   \parindent0pt\relax
7905   \textbf{\glsnavigation}\par\indexspace}%
```

Put a hypertarget at the start of each group

```
7906 \renewcommand*{\glsgroupheading}[1]{%
7907   \par
7908   \def\@gls@prevlevel{-1}%
7909   \hangindent0pt\relax
7910   \parindent0pt\relax
7911   \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
7912   \indexspace}}
```

5.7 Glossary Styles using supertabular environment (glossary-super package)

The glossary styles defined in the package use the supertabular environment.

```
7913 \ProvidesPackage{glossary-super}[2013/11/14 v4.0 (NLCT)]
```

Requires the package:

```
7914 \RequirePackage{supertabular}
```

`\glsdescwidth` This is a length that governs the width of the description column. This may already have been defined if has been loaded.

```
7915 \@ifundefined{glsdescwidth}{%
7916   \newlength{glsdescwidth
7917   \setlength{glsdescwidth}{0.6\hsize}
7918 }{}}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined if has been loaded.

```
7919 \@ifundefined{glspagelistwidth}{%
7920   \newlength{glspagelistwidth
7921   \setlength{glspagelistwidth}{0.1\hsize}
7922 }{}}
```

`super` The super glossary style uses the supertabular environment (it uses lengths defined in the package.)

```
7923 \newglossarystyle{super}{%
```

Put the glossary in a supertabular environment with two columns and no head or tail:

```
7924   \renewenvironment{theglossary}{%
7925     {\tablehead{}\tabletail}%
7926     \begin{supertabular}{lp{glsdescwidth}}%
7927     {\end{supertabular}}%
```

Do nothing at the start of the table:

```
7928 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7929 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries put in a row (name in first column, description and page list in second column):

```
7930 \renewcommand{\glossentry}[2]{%
7931   \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7932   \glossentrydesc{##1}\glspostdescription\space ##2\tabularnewline
7933 }%
```

Sub entries put in a row (no name, description and page list in second column):

```
7934 \renewcommand{\subglossentry}[3]{%
7935   &
7936   \glssubentryitem{##2}%
7937   \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space
7938   ##3\tabularnewline
7939 }%
```

Blank row between groups:

```
7940 \renewcommand*{\glsgroupskip}{}%
7941   \ifglsnogroupskip\else & \tabularnewline\fi}%
7942 }
```

superborder The superborder style is like the above, but with horizontal and vertical lines:

```
7943 \newglossarystyle{superborder}{}%
```

Base it on the `glostylesuper` style:

```
7944 \setglossarystyle{super}{}%
```

Put the glossary in a `supertabular` environment with two columns and a horizontal line in the head and tail:

```
7945 \renewenvironment{theglossary}%
7946   {\tablehead{\hline}\tabletail{\hline}%
7947   \begin{supertabular}{|l|p{\glsdescwidth}|}}%
7948   {\end{supertabular}}%
7949 }
```

superheader The superheader style is like the super style, but with a header:

```
7950 \newglossarystyle{superheader}{}%
```

Base it on the `glostylesuper` style:

```
7951 \setglossarystyle{super}{}%
```

Put the glossary in a `supertabular` environment with two columns, a header and no tail:

```
7952 \renewenvironment{theglossary}%
7953   {\tablehead{\bfseries \entryname &
```

```

7954 \bfseries\descriptionname\tabularnewline}%
7955 \tabletail{}}%
7956 \begin{supertabular}{lp{\glsgdescwidth}}}%
7957 {\end{supertabular}}}%
7958 }

```

superheaderborder The superheaderborder style is like the super style but with a header and border:

```
7959 \newglossarystyle{superheaderborder}{%
```

Base it on the glostylesuper style:

```
7960 \setglossarystyle{super}%
```

Put the glossary in a supertabular environment with two columns, a header and horizontal lines above and below the table:

```

7961 \renewenvironment{theglossary}%
7962 {\tablehead{\hline\bfseries \entryname &
7963 \bfseries \descriptionname\tabularnewline\hline}%
7964 \tabletail{\hline}
7965 \begin{supertabular}{|lp{\glsgdescwidth}|}%
7966 {\end{supertabular}}}%
7967 }

```

super3col The super3col style is like the super style, but with 3 columns:

```
7968 \newglossarystyle{super3col}{%
```

Put the glossary in a supertabular environment with three columns and no head or tail:

```

7969 \renewenvironment{theglossary}%
7970 {\tablehead{}\tabletail{}}%
7971 \begin{supertabular}{lp{\glsgdescwidth}p{\glspagelistwidth}}}%
7972 {\end{supertabular}}}%

```

Do nothing at the start of the table:

```
7973 \renewcommand*\glossaryheader{}}%
```

No group headings:

```
7974 \renewcommand*\glsgroupheading[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```

7975 \renewcommand{\glossentry}[2]{%
7976 \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7977 \glossentrydesc{##1} & ##2\tabularnewline
7978 }%

```

Sub entries on a row (no name, description in second column, page list in last column):

```

7979 \renewcommand{\subglossentry}[3]{%
7980 &
7981 \glssubentryitem{##2}%
7982 \glstarget{##2}{\strut}\glossentrydesc{##2} &

```

```

7983     ##3\tabularnewline
7984 }%

```

Blank row between groups:

```

7985 \renewcommand*{\glsgroupskip}{%
7986 \ifglsgnigroupskip\else & &\tabularnewline\fi}%
7987 }

```

super3colborder The super3colborder style is like the super3col style, but with a border:

```

7988 \newglossarystyle{super3colborder}{%
    Base it on the glostylesuper3col style:
7989 \setglossarystyle{super3col}%
    Put the glossary in a supertabular environment with three columns and a horizontal line in the head and tail:
7990 \renewenvironment{theglossary}%
7991 {\tablehead{\hline}\tabletail{\hline}%
7992 \begin{supertabular}{|l|p{\glsgdescwidth}|p{\glspagelistwidth}|}%
7993 {\end{supertabular}}%
7994 }

```

super3colheader The super3colheader style is like the super3col style but with a header row:

```

7995 \newglossarystyle{super3colheader}{%
    Base it on the glostylesuper3col style:
7996 \setglossarystyle{super3col}%
    Put the glossary in a supertabular environment with three columns, a header and no tail:
7997 \renewenvironment{theglossary}%
7998 {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
7999 \bfseries\pagelistname\tabularnewline}\tabletail{}}%
8000 \begin{supertabular}{lp{\glsgdescwidth}p{\glspagelistwidth}}%
8001 {\end{supertabular}}%
8002 }

```

super3colheaderborder The super3colheaderborder style is like the super3col style but with a header and border:

```

8003 \newglossarystyle{super3colheaderborder}{%
    Base it on the glostylesuper3colborder style:
8004 \setglossarystyle{super3colborder}%
    Put the glossary in a supertabular environment with three columns, a header with horizontal lines and a horizontal line in the tail:
8005 \renewenvironment{theglossary}%
8006 {\tablehead{\hline
8007 \bfseries\entryname&\bfseries\descriptionname&
8008 \bfseries\pagelistname\tabularnewline\hline}%
8009 \tabletail{\hline}%

```

```

8010     \begin{supertabular}{|l|p{\glstdescwidth}|p{\glspagelistwidth}|}%
8011     {\end{supertabular}}%
8012 }

```

super4col The `super4col` glossary style has four columns, where the third column contains the value of the corresponding symbol key used when that entry was defined.

```
8013 \newglossarystyle{super4col}{%
```

Put the glossary in a `supertabular` environment with four columns and no head or tail:

```

8014   \renewenvironment{theglossary}%
8015   {\tablehead{}\tabletail}%
8016   \begin{supertabular}{|l|l|l|l|}%
8017   \end{supertabular}}%

```

Do nothing at the start of the table:

```
8018   \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
8019   \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row with the name in the first column, description in second column, symbol in third column and page list in last column:

```

8020   \renewcommand{\glossentry}[2]{%
8021     \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
8022     \glossentrydesc{##1} &
8023     \glossentrysymbol{##1} & ##3\tabularnewline
8024   }%

```

Sub entries on a row with no name, the description in the second column, symbol in third column and page list in last column:

```

8025   \renewcommand{\subglossentry}[3]{%
8026     &
8027     \glssubentryitem{##2}%
8028     \glstarget{##2}{\strut}\glossentrydesc{##2} &
8029     \glossentrysymbol{##2} & ##3\tabularnewline
8030   }%

```

Blank row between groups:

```

8031   \renewcommand*{\glsgroupskip}{%
8032     \ifglsnogroupskip\else & & \tabularnewline\fi}%
8033 }

```

super4colheader The `super4colheader` style is like the `super4col` but with a header row.

```
8034 \newglossarystyle{super4colheader}{%
```

Base it on the `glostylesuper4col` style:

```
8035   \setglossarystyle{super4col}%
```

Put the glossary in a `supertabular` environment with four columns, a header and no tail:

```

8036 \renewenvironment{theglossary}%
8037   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
8038     \bfseries\symbolname &
8039     \bfseries\pagelistname\tabularnewline}%
8040   \tabletail{}}%
8041   \begin{supertabular}{|l|l|l|l|}%
8042   {\end{supertabular}}}%
8043 }

```

super4colborder The super4colborder style is like the super4col but with a border.

```

8044 \newglossarystyle{super4colborder}{%
      Base it on the glostylesuper4col style:
8045   \setglossarystyle{super4col}%
      Put the glossary in a supertabular environment with four columns and a horizontal line in the head and tail:
8046   \renewenvironment{theglossary}%
8047     {\tablehead{\hline}\tabletail{\hline}%
8048     \begin{supertabular}{|l|l|l|l|}%
8049     {\end{supertabular}}}%
8050 }

```

per4colheaderborder The super4colheaderborder style is like the super4col but with a header and border.

```

8051 \newglossarystyle{super4colheaderborder}{%
      Base it on the glostylesuper4col style:
8052   \setglossarystyle{super4col}%
      Put the glossary in a supertabular environment with four columns and a header bordered by horizontal lines and a horizontal line in the tail:
8053   \renewenvironment{theglossary}%
8054     {\tablehead{\hline\bfseries\entryname&\bfseries\descriptionname&
8055       \bfseries\symbolname &
8056       \bfseries\pagelistname\tabularnewline\hline}%
8057     \tabletail{\hline}%
8058     \begin{supertabular}{|l|l|l|l|}%
8059     {\end{supertabular}}}%
8060 }

```

altsuper4col The altsuper4col glossary style is like super4col but has provision for multiline descriptions.

```

8061 \newglossarystyle{altsuper4col}{%
      Base it on the glostylesuper4col style:
8062   \setglossarystyle{super4col}%
      Put the glossary in a supertabular environment with four columns and no head or tail:

```



```

8063 \renewenvironment{theglossary}%
8064   {\tablehead{}\tabletail{}}%
8065   \begin{supertabular}{lp{\glstdescwidth}lp{\glspagelistwidth}}}%
8066   {\end{supertabular}}}%
8067 }

```

altsuper4colheader The altsuper4colheader style is like the altsuper4col but with a header row.

```

8068 \newglossarystyle{altsuper4colheader}{%
      Base it on the glostylesuper4colheader style:
8069   \setglossarystyle{super4colheader}%
      Put the glossary in a supertabular environment with four columns, a header and
      no tail:
8070   \renewenvironment{theglossary}%
8071     {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
8072               \bfseries\symbolname &
8073               \bfseries\pagelistname\tabularnewline}\tabletail{}}%
8074     \begin{supertabular}{lp{\glstdescwidth}lp{\glspagelistwidth}}}%
8075     {\end{supertabular}}}%
8076 }

```

altsuper4colborder The altsuper4colborder style is like the altsuper4col but with a border.

```

8077 \newglossarystyle{altsuper4colborder}{%
      Base it on the glostylesuper4colborder style:
8078   \setglossarystyle{super4colborder}%
      Put the glossary in a supertabular environment with four columns and a hori-
      zontal line in the head and tail:
8079   \renewenvironment{theglossary}%
8080     {\tablehead{\hline}\tabletail{\hline}%
8081     \begin{supertabular}%
8082       {\lllp{\glstdescwidth}|lp{\glspagelistwidth}|}}%
8083     {\end{supertabular}}}%
8084 }

```

per4colheaderborder The altsuper4colheaderborder style is like the altsuper4col but with a header and border.

```

8085 \newglossarystyle{altsuper4colheaderborder}{%
      Base it on the glostylesuper4colheaderborder style:
8086   \setglossarystyle{super4colheaderborder}%
      Put the glossary in a supertabular environment with four columns and a header
      bordered by horizontal lines and a horizontal line in the tail:
8087   \renewenvironment{theglossary}%
8088     {\tablehead{\hline
8089               \bfseries\entryname &
8090               \bfseries\descriptionname &
8091               \bfseries\symbolname &

```

```

8092      \bfseries\pagelistname\tabularnewline\hline}%
8093      \tabletail{\hline}%
8094      \begin{supertabular}%
8095          {\lllp{\glsdescwidth}\lllp{\glspagelistwidth}\l}%
8096      {\end{supertabular}}%
8097 }

```

5.8 Glossary Styles using supertabular environment (glossary-superragged package)

The glossary styles defined in the package use the supertabular environment. These styles are like those provided by the package, except that the multiline columns have ragged right justification.

```

8098 \ProvidesPackage{glossary-superragged}[2013/11/14 v4.0 (NLCT)]

```

Requires the package:

```

8099 \RequirePackage{array}

```

Requires the package:

```

8100 \RequirePackage{supertabular}

```

`\glsdescwidth` This is a length that governs the width of the description column. This may already have been defined.

```

8101 \@ifundefined{glsdescwidth}{%
8102   \newlength\glsdescwidth
8103   \setlength{\glsdescwidth}{0.6\hsize}
8104 }{}

```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

```

8105 \@ifundefined{glspagelistwidth}{%
8106   \newlength\glspagelistwidth
8107   \setlength{\glspagelistwidth}{0.1\hsize}
8108 }{}

```

`superragged` The superragged glossary style uses the supertabular environment.

```

8109 \newglossarystyle{superragged}{%

```

Put the glossary in a supertabular environment with two columns and no head or tail:

```

8110   \renewenvironment{theglossary}%
8111       {\tablehead{}\tabletail{}}%
8112       \begin{supertabular}{l>{\raggedright}p{\glsdescwidth}}%
8113       {\end{supertabular}}%

```

Do nothing at the start of the table:

```

8114   \renewcommand*{\glossaryheader}{}%

```

No group headings:

```

8115   \renewcommand*{\glsgroupheading}[1]{}%

```

Main (level 0) entries put in a row (name in first column, description and page list in second column):

```

8116 \renewcommand{\glossentry}[2]{%
8117   \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
8118   \glossentrydesc{##1}\glspostdescription\space ##2%
8119   \tabularnewline
8120 }%

```

Sub entries put in a row (no name, description and page list in second column):

```

8121 \renewcommand{\subglossentry}[3]{%
8122   &
8123   \glssubentryitem{##2}%
8124   \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space
8125   ##3%
8126   \tabularnewline
8127 }%

```

Blank row between groups:

```

8128 \renewcommand*{\glsgroupskip}{\ifglsgnোগroupskip\else & \tabularnewline\fi}%
8129 }

```

superraggedborder The superraggedborder style is like the above, but with horizontal and vertical lines:

```

8130 \newglossarystyle{superraggedborder}{%

```

Base it on the glostylessuperragged style:

```

8131 \setglossarystyle{superragged}%

```

Put the glossary in a supertabular environment with two columns and a horizontal line in the head and tail:

```

8132 \renewenvironment{theglossary}%
8133   {\tablehead{\hline}\tabletail{\hline}%
8134   \begin{supertabular}{|l|>\raggedright}p{\glsgdescwidth}}}%
8135   {\end{supertabular}}%
8136 }

```

superraggedheader The superraggedheader style is like the super style, but with a header:

```

8137 \newglossarystyle{superraggedheader}{%

```

Base it on the glostylessuperragged style:

```

8138 \setglossarystyle{superragged}%

```

Put the glossary in a supertabular environment with two columns, a header and no tail:

```

8139 \renewenvironment{theglossary}%
8140   {\tablehead{\bfseries \entryname & \bfseries \descriptionname
8141     \tabularnewline}%
8142   \tabletail{}}%
8143   \begin{supertabular}{|l|>\raggedright}p{\glsgdescwidth}}}%
8144   {\end{supertabular}}%
8145 }

```

rraggedheaderborder The superraggedheaderborder style is like the superragged style but with a header and border:

```
8146 \newglossarystyle{superraggedheaderborder}{%
```

Base it on the glostylesuper style:

```
8147 \setglossarystyle{superragged}%
```

Put the glossary in a supertabular environment with two columns, a header and horizontal lines above and below the table:

```
8148 \renewenvironment{theglossary}%  
8149 {\tablehead{\hline\bfseries \entryname &  
8150 \bfseries \descriptionname\tabularnewline\hline}%  
8151 \tabletail{\hline}  
8152 \begin{supertabular}{|l|>\raggedright}p{\glsgdescwidth}|}%  
8153 {\end{supertabular}}%  
8154 }
```

superragged3col The superragged3col style is like the superragged style, but with 3 columns:

```
8155 \newglossarystyle{superragged3col}{%
```

Put the glossary in a supertabular environment with three columns and no head or tail:

```
8156 \renewenvironment{theglossary}%  
8157 {\tablehead{}\tabletail{}}%  
8158 \begin{supertabular}{|l>\raggedright}p{\glsgdescwidth}%  
8159 >\raggedright}p{\glspagelistwidth}}}%  
8160 {\end{supertabular}}%
```

Do nothing at the start of the table:

```
8161 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
8162 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
8163 \renewcommand{\glossentry}[2]{%  
8164 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &  
8165 \glossentrydesc{##1} &  
8166 ##2\tabularnewline  
8167 }%
```

Sub entries on a row (no name, description in second column, page list in last column):

```
8168 \renewcommand{\subglossentry}[3]{%  
8169 &  
8170 \glssubentryitem{##2}%  
8171 \glstarget{##2}{\strut}\glossentrydesc{##2} &  
8172 ##3\tabularnewline  
8173 }%
```

Blank row between groups:

```
8174 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & &\tabularnewline\fi}%
8175 }
```

superragged3colborder The `superragged3colborder` style is like the `superragged3col` style, but with a border:

```
8176 \newglossarystyle{superragged3colborder}{%
```

Base it on the `glostylesuperragged3col` style:

```
8177 \setglossarystyle{superragged3col}%
```

Put the glossary in a `supertabular` environment with three columns and a horizontal line in the head and tail:

```
8178 \renewenvironment{theglossary}%
8179 {\tablehead{\hline}\tabletail{\hline}%
8180 \begin{supertabular}{|l|>{\raggedright}p{\glsdescwidth}|%
8181 >{\raggedright}p{\glspagelistwidth}|}%
8182 {\end{supertabular}}%
8183 }
```

superragged3colheader The `superragged3colheader` style is like the `superragged3col` style but with a header row:

```
8184 \newglossarystyle{superragged3colheader}{%
```

Base it on the `glostylesuperragged3col` style:

```
8185 \setglossarystyle{superragged3col}%
```

Put the glossary in a `supertabular` environment with three columns, a header and no tail:

```
8186 \renewenvironment{theglossary}%
8187 {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
8188 \bfseries\pagelistname\tabularnewline}\tabletail{}}%
8189 \begin{supertabular}{|l>{\raggedright}p{\glsdescwidth}%
8190 >{\raggedright}p{\glspagelistwidth}}}%
8191 {\end{supertabular}}%
8192 }
```

superragged3colheaderborder The `superragged3colheaderborder` style is like the `superragged3col` style but with a header and border:

```
8193 \newglossarystyle{superragged3colheaderborder}{%
```

Base it on the `glostylesuperragged3colborder` style:

```
8194 \setglossarystyle{superragged3colborder}%
```

Put the glossary in a `supertabular` environment with three columns, a header with horizontal lines and a horizontal line in the tail:

```
8195 \renewenvironment{theglossary}%
8196 {\tablehead{\hline
8197 \bfseries\entryname&\bfseries\descriptionname&
8198 \bfseries\pagelistname\tabularnewline\hline}%

```

```

8199     \tabletail{\hline}%
8200     \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}|%
8201       >{\raggedright}p{\glspagelistwidth}|}%
8202     {\end{supertabular}}%
8203 }

```

`altsuperragged4col` The `altsuperragged4col` glossary style is like `altsuper4col` style in the package but uses ragged right formatting in the description and page list columns.

```

8204 \newglossarystyle{altsuperragged4col}{%

```

Put the glossary in a `supertabular` environment with four columns and no head or tail:

```

8205   \renewenvironment{theglossary}%
8206     {\tablehead{}\tabletail{}}%
8207     \begin{supertabular}{|l>{\raggedright}p{\glsgdescwidth}l%
8208       >{\raggedright}p{\glspagelistwidth}|}%
8209     {\end{supertabular}}%

```

Do nothing at the start of the table:

```

8210   \renewcommand*{\glossaryheader}{}%

```

No group headings:

```

8211   \renewcommand*{\glsgroupheading}[1]{}%

```

Main (level 0) entries on a row with the name in the first column, description in second column, symbol in third column and page list in last column:

```

8212   \renewcommand{\glossentry}[2]{%
8213     \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
8214     \glossentrydesc{##1} &
8215     \glossentrysymbol{##1} & ##2\tabularnewline
8216   }%

```

Sub entries on a row with no name, the description in the second column, symbol in third column and page list in last column:

```

8217   \renewcommand{\subglossentry}[3]{%
8218     &
8219     \glssubentryitem{##2}%
8220     \glstarget{##2}{\strut}\glossentrydesc{##2} &
8221     \glossentrysymbol{##2} & ##3\tabularnewline
8222   }%

```

Blank row between groups:

```

8223   \renewcommand*{\glsgroupskip}{\ifglsgnোগroupskip\else & & \tabularnewline\fi}%
8224 }

```

`perragged4colheader` The `altsuperragged4colheader` style is like the `altsuperragged4col` style but with a header row.

```

8225 \newglossarystyle{altsuperragged4colheader}{%

```

Base it on the `glostylealtsuperragged4col` style:

```

8226   \setglossarystyle{altsuperragged4col}%

```

Put the glossary in a supertabular environment with four columns, a header and no tail:

```

8227 \renewenvironment{theglossary}%
8228   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
8229     \bfseries\symbolname &
8230     \bfseries\pagelistname\tabularnewline}\tabletail{}}%
8231   \begin{supertabular}{1>{\raggedright}p{\glsdescwidth}1%
8232     >{\raggedright}p{\glspagelistwidth}}}%
8233   {\end{supertabular}}%
8234 }

```

`altperragged4colborder` The `altperragged4colborder` style is like the `altperragged4col` style but with a border.

```

8235 \newglossarystyle{altperragged4colborder}{%

```

Base it on the `glostylealtperragged4col` style:

```

8236 \setglossarystyle{altperragged4col}%

```

Put the glossary in a supertabular environment with four columns and a horizontal line in the head and tail:

```

8237 \renewenvironment{theglossary}%
8238   {\tablehead{\hline}\tabletail{\hline}%
8239   \begin{supertabular}%
8240     {|1|>{\raggedright}p{\glsdescwidth}|1|%
8241     >{\raggedright}p{\glspagelistwidth}|}%
8242   {\end{supertabular}}%
8243 }

```

`altperragged4colheaderborder` The `altperragged4colheaderborder` style is like the `altperragged4col` style but with a header and border.

```

8244 \newglossarystyle{altperragged4colheaderborder}{%

```

Base it on the `glostylealtperragged4col` style:

```

8245 \setglossarystyle{altperragged4col}%

```

Put the glossary in a supertabular environment with four columns and a header bordered by horizontal lines and a horizontal line in the tail:

```

8246 \renewenvironment{theglossary}%
8247   {\tablehead{\hline
8248     \bfseries\entryname &
8249     \bfseries\descriptionname &
8250     \bfseries\symbolname &
8251     \bfseries\pagelistname\tabularnewline\hline}%
8252   \tabletail{\hline}%
8253   \begin{supertabular}%
8254     {|1|>{\raggedright}p{\glsdescwidth}|1|%
8255     >{\raggedright}p{\glspagelistwidth}|}%
8256   {\end{supertabular}}%
8257 }

```

5.9 Tree Styles (glossary-tree.sty)

The style file defines glossary styles that have a tree-like structure. These are designed for hierarchical glossaries.

8258 \ProvidesPackage{glossary-tree}[2014/07/30 v4.08 (NLCT)]

\glstreenamefmt Format used to display the name in the tree styles. (This may be counteracted by \glsnamefont.) This command is also used to format the group headings.

8259 \newcommand*{\glstreenamefmt}[1]{\textbf{#1}}

index The index glossary style is similar in style to the way indices are usually typeset using \item, \subitem and \subsubitem. The entry name is set in bold. If an entry has a symbol, it is placed in brackets after the name. Then the description is displayed, followed by the number list. This style allows up to three levels.

8260 \newglossarystyle{index}{%

Set the paragraph indentation and skip and define \item to be the same as that used by theindex:

8261 \renewenvironment{theglossary}{%
8262 {\setlength{\parindent}{0pt}}%
8263 \setlength{\parskip}{0pt plus 0.3pt}}%
8264 \let\item\@idxitem}%

8265 {\par}%

Do nothing at the start of the environment:

8266 \renewcommand*{\glossaryheader}{}%

No group headers:

8267 \renewcommand*{\glsgroupheading}[1]{}%

Main (level 0) entry starts a new item with the name in bold followed by the symbol in brackets (if it exists), the description and the page list.

8268 \renewcommand*{\glossentry}[2]{%
8269 \item\glstentryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
8270 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
8271 \space \glossentrydesc{##1}\glspostdescription\space ##2%
8272 }%

Sub entries: level 1 entries use \subitem, levels greater than 1 use \subsubitem.

The level (##1) shouldn't be 0, as that's catered by \glossentry, but for completeness, if the level is 0, \item is used. The name is put in bold, followed by the symbol in brackets (if it exists), the description and the page list.

8273 \renewcommand{\subglossentry}[3]{%
8274 \ifcase##1\relax
8275 % level 0
8276 \item
8277 \or
8278 % level 1
8279 \subitem


```

8280     \glssubentryitem{##2}%
8281     \else
8282     % all other levels
8283     \subsubitem
8284     \fi
8285     \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}%
8286     \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
8287     \space\glossentrydesc{##2}\glspostdescription\space ##3%
8288 }%

```

Vertical gap between groups is the same as that used by indices:

```

8289 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}

```

indexgroup The indexgroup style is like the index style but has headings.

```

8290 \newglossarystyle{indexgroup}{%

```

Base it on the glostyleindex style:

```

8291 \setglossarystyle{index}%

```

Add a heading for each group. This puts the group's title in bold followed by a vertical gap.

```

8292 \renewcommand*{\glsgroupheading}[1]{%
8293 \item\glstreenamefmt{\glsgetgrouptitle{##1}}\indexspace}%
8294 }

```

indexhypergroup The indexhypergroup style is like the indexgroup style but has hyper navigation.

```

8295 \newglossarystyle{indexhypergroup}{%

```

Base it on the glostyleindex style:

```

8296 \setglossarystyle{index}%

```

Put navigation links to the groups at the start of the glossary:

```

8297 \renewcommand*{\glossaryheader}{%
8298 \item\glstreenamefmt{\glslnavigation}\indexspace}%

```

Add a heading for each group (with a target). The group's title is in bold followed by a vertical gap.

```

8299 \renewcommand*{\glsgroupheading}[1]{%
8300 \item\glstreenamefmt{\glslnavhypertarget{##1}{\glsggetgrouptitle{##1}}}%
8301 \indexspace}%
8302 }

```

tree The tree glossary style is similar in style to the index style, but can have arbitrary levels.

```

8303 \newglossarystyle{tree}{%

```

Set the paragraph indentation and skip:

```

8304 \renewenvironment{theglossary}%
8305 {\setlength{\parindent}{0pt}%
8306 \setlength{\parskip}{0pt plus 0.3pt}}%
8307 {}%

```

Do nothing at the start of the theglossary environment:

```
8308 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
8309 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries: name in bold, followed by symbol in brackets (if it exists), the description and the page list:

```
8310 \renewcommand{\glossentry}[2]{%
8311   \hangindent0pt\relax
8312   \parindent0pt\relax
8313   \glstryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
8314   \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
8315   \space\glossentrydesc{##1}\glspostdescription\space##2\par
8316 }%
```

Sub entries: level $\langle n \rangle$ is indented by $\langle n \rangle$ times `\glstreeindent`. The name is in bold, followed by the symbol in brackets (if it exists), the description and the page list.

```
8317 \renewcommand{\subglossentry}[3]{%
8318   \hangindent##1\glstreeindent\relax
8319   \parindent##1\glstreeindent\relax
8320   \ifnum##1=1\relax
8321     \glssubentryitem{##2}%
8322   \fi
8323   \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}%
8324   \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
8325   \space\glossentrydesc{##2}\glspostdescription\space ##3\par
8326 }%
```

Vertical gap between groups is the same as that used by indices:

```
8327 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}}
```

treegroup Like the tree style but the glossary groups have headings.

```
8328 \newglossarystyle{treegroup}{%
```

Base it on the glostyletree style:

```
8329 \setglossarystyle{tree}%
```

Each group has a heading (in bold) followed by a vertical gap):

```
8330 \renewcommand{\glsgroupheading}[1]{\par
8331   \noindent\glstreenamefmt{\glsgrouptitle{##1}}\par\indexspace}%
8332 }
```

treehypergroup The treehypergroup style is like the treegroup style, but has a set of links to the groups at the start of the glossary.

```
8333 \newglossarystyle{treehypergroup}{%
```

Base it on the glostyletree style:

```
8334 \setglossarystyle{tree}%
```

Put navigation links to the groups at the start of the theglossary environment:

```
8335 \renewcommand*{\glossaryheader}{%
8336 \par\noindent\glstreenamefmt{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
8337 \renewcommand*{\glsgroupheading}[1]{%
8338 \par\noindent
8339 \glstreenamefmt{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}\par
8340 \indexspace}%
8341 }
```

`\glstreeindent` Length governing left indent for each level of the tree style.

```
8342 \newlength\glstreeindent
8343 \setlength{\glstreeindent}{10pt}
```

`treenoname` The `treenoname` glossary style is like the tree style, but doesn't print the name or symbol for sub-levels.

```
8344 \newglossarystyle{treenoname}{%
```

Set the paragraph indentation and skip:

```
8345 \renewenvironment{theglossary}%
8346 {\setlength{\parindent}{0pt}%
8347 \setlength{\parskip}{0pt plus 0.3pt}}%
8348 {}%
```

No header:

```
8349 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
8350 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries: the name is in bold, followed by the symbol in brackets (if it exists), the description and the page list.

```
8351 \renewcommand{\glossentry}[2]{%
8352 \hangindent0pt\relax
8353 \parindent0pt\relax
8354 \glstryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
8355 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
8356 \space\glossentrydesc{##1}\glspostdescription\space##2\par
8357 }%
```

Sub entries: level $\langle n \rangle$ is indented by $\langle n \rangle$ times `\glstreeindent`. The name and symbol are omitted. The description followed by the page list are displayed.

```
8358 \renewcommand{\subglossentry}[3]{%
8359 \hangindent##1\glstreeindent\relax
8360 \parindent##1\glstreeindent\relax
8361 \ifnum##1=1\relax
8362 \glssubentryitem{##2}%
8363 \fi
8364 \glstarget{##2}{\strut}%
8365 \glossentrydesc{##2}\glspostdescription\space##3\par
8366 }%
```

Vertical gap between groups is the same as that used by indices:

```
8367 \renewcommand*{\glsgroupskip}{\ifglsgroupskip\else\indexspace\fi}%
8368 }
```

treenonamegroup Like the `treenoname` style but the glossary groups have headings.

```
8369 \newglossarystyle{treenonamegroup}{%
```

Base it on the `glostyletreenoname` style:

```
8370 \setglossarystyle{treenoname}%
```

Give each group a heading:

```
8371 \renewcommand{\glsgroupheading}[1]{\par
```

```
8372 \noindent\glstreenamfmt{\glsgrouptitle{##1}}\par\indexspace}%
```

```
8373 }
```

treenonamehypergroup The `treenonamehypergroup` style is like the `treenonamegroup` style, but has a set of links to the groups at the start of the glossary.

```
8374 \newglossarystyle{treenonamehypergroup}{%
```

Base it on the `glostyletreenoname` style:

```
8375 \setglossarystyle{treenoname}%
```

Put navigation links to the groups at the start of the theglossary environment:

```
8376 \renewcommand*{\glossaryheader}{%
```

```
8377 \par\noindent\glstreenamfmt{\glsgroupnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
8378 \renewcommand*{\glsgroupheading}[1]{%
```

```
8379 \par\noindent
```

```
8380 \glstreenamfmt{\glsgrouphypertarget{##1}{\glsgrouptitle{##1}}}\par
```

```
8381 \indexspace}%
```

```
8382 }
```

\glissetwidest `\glissetwidest[level]{text}` sets the widest text for the given level. It is used by the `alttree` glossary styles to determine the indentation of each level.

```
8383 \newcommand*{\glissetwidest}[2][0]{%
```

```
8384 \expandafter\def\csname @glswidestname\romannumeral#1\endcsname{%
```

```
8385 #2}%
```

```
8386 }
```

\@glswidestname Initialise `\@glswidestname`.

```
8387 \newcommand*{\@glswidestname}{}
```

alttree The `alttree` glossary style is similar in style to the `tree` style, but the indentation is obtained from the width of `\@glswidestname` which is set using `\glissetwidest`.

```
8388 \newglossarystyle{alttree}{%
```

Redefine theglossary environment.

```
8389 \renewenvironment{theglossary}%  
8390   {\def\@gls@prevlevel{-1}%  
8391    \mbox{}\par}%  
8392   {\par}%
```

Set the header and group headers to nothing.

```
8393 \renewcommand*\glossaryheader{}%  
8394 \renewcommand*\glsgroupheading[1]{}%
```

Redefine the way that the level 0 entries are displayed.

```
8395 \renewcommand{\glosentry}[2]{%  
8396   \ifnum\@gls@prevlevel=0\relax  
8397   \else
```

Find out how big the indentation should be by measuring the widest entry.

```
8398     \settowidth{\glstreeindent}{\glstreenamfmt{\@glswidestname\space}}%  
8399   \fi
```

Set the hangindent and paragraph indent.

```
8400     \hangindent\glstreeindent  
8401     \parindent\glstreeindent
```

Put the name to the left of the paragraph block.

```
8402     \makebox[0pt][r]{\makebox[\glstreeindent][l]{%  
8403       \glstryitem{##1}\glstreenamfmt{\glstarget{##1}{\glosentryname{##1}}}}}%
```

If the symbol is missing, ignore it, otherwise put it in brackets.

```
8404     \ifglshassymbol{##1}{\space(\glosentrysymbol{##1})}{}%
```

Do the description followed by the description terminator and location list.

```
8405     \glosentrydesc{##1}\glspostdescription \space ##2\par
```

Set the previous level to 0.

```
8406     \def\@gls@prevlevel{0}%  
8407   }%
```

Redefine the way sub-entries are displayed.

```
8408 \renewcommand{\subglosentry}[3]{%
```

Increment and display the sub-entry counter if this is a level 1 entry and the sub-entry counter is in use.

```
8409     \ifnum##1=1\relax  
8410     \glssubentryitem{##2}%  
8411   \fi
```

If the level hasn't changed, keep the same settings, otherwise adjust \glstreeindent accordingly.

```
8412     \ifnum\@gls@prevlevel=##1\relax  
8413     \else
```

Compute the widest entry for this level, or for level 0 if not defined for this level.
Store in \gls@tmplen

```

8414     \@ifundefined{@glswidestname\romannumeral##1}{%
8415         \settowidth{\gls@tmplen}{\glstreenamefmt{\@glswidestname\space}}}%
8416         \settowidth{\gls@tmplen}{\glstreenamefmt{%
8417             \csname @glswidestname\romannumeral##1\endcsname\space}}}%

```

Determine if going up or down a level

```

8418     \ifnum\@gls@prevlevel<##1\relax

```

Depth has increased, so add the width of the widest entry to \glstreeindent.

```

8419         \setlength\glstreeindent\gls@tmplen
8420         \addtolength\glstreeindent\parindent
8421         \parindent\glstreeindent
8422     \else

```

Depth has decreased, so subtract width of the widest entry from the previous level to \glstreeindent. First determine the width of the widest entry for the previous level and store in \glstreeindent.

```

8423         \@ifundefined{@glswidestname\romannumeral\@gls@prevlevel}{%
8424             \settowidth{\glstreeindent}{\glstreenamefmt{%
8425                 \@glswidestname\space}}}%
8426             \settowidth{\glstreeindent}{\glstreenamefmt{%
8427                 \csname @glswidestname\romannumeral\@gls@prevlevel
8428                     \endcsname\space}}}%

```

Subtract this length from the previous level's paragraph indent and set to \glstreeindent.

```

8429         \addtolength\parindent{-\glstreeindent}%
8430         \setlength\glstreeindent\parindent
8431     \fi
8432 \fi

```

Set the hanging indentation.

```

8433     \hangindent\glstreeindent

```

Put the name to the left of the paragraph block

```

8434     \makebox[0pt][r]{\makebox[\gls@tmplen][l]{%
8435         \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}}}%

```

If the symbol is missing, ignore it, otherwise put it in brackets.

```

8436     \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%

```

Do the description followed by the description terminator and location list.

```

8437     \glossentrydesc{##2}\glspostdescription\space ##3\par

```

Set the previous level macro to the current level.

```

8438     \def\@gls@prevlevel{##1}%
8439 }%

```

Vertical gap between groups is the same as that used by indices:

```

8440 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
8441 }

```

alttreegroup Like the `alttree` style but the glossary groups have headings.

```
8442 \newglossarystyle{alttreegroup}{%  
    Base it on the glostylealttree style:  
8443 \setglossarystyle{alttree}%  
    Give each group a heading.  
8444 \renewcommand{\glsgroupheading}[1]{\par  
8445 \def\@gls@prevlevel{-1}%  
8446 \hangindent0pt\relax  
8447 \parindent0pt\relax  
8448 \glstreenamfmt{\glsgetgrouptitle{##1}}\par\indexspace}%  
8449 }
```

alttreehypergroup The `alttreehypergroup` style is like the `alttreegroup` style, but has a set of links to the groups at the start of the glossary.

```
8450 \newglossarystyle{alttreehypergroup}{%  
    Base it on the glostylealttree style:  
8451 \setglossarystyle{alttree}%  
    Put the navigation links in the header  
8452 \renewcommand*\glossaryheader{%  
8453 \par  
8454 \def\@gls@prevlevel{-1}%  
8455 \hangindent0pt\relax  
8456 \parindent0pt\relax  
8457 \glstreenamfmt{\glsnavigation}\par\indexspace}%  
    Put a hypertarget at the start of each group  
8458 \renewcommand*\glsgroupheading[1]{%  
8459 \par  
8460 \def\@gls@prevlevel{-1}%  
8461 \hangindent0pt\relax  
8462 \parindent0pt\relax  
8463 \glstreenamfmt{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par  
8464 \indexspace}}
```

6 glossaries-compatible-207

Provides compatibility with version 2.07 and below. This uses original `glossaries` `xindy` and `makeindex` formatting, so can be used with old documents that had customized style files, but hyperlinks may not work properly.

```
8465 \NeedsTeXFormat{LaTeX2e}  
8466 \ProvidesPackage{glossaries-compatible-207}[2011/04/02 v1.0 (NLCT)]
```

\GlsAddXdyAttribute Adds an attribute in old format.

```
8467 \ifglsxindy  
8468 \renewcommand*\GlsAddXdyAttribute[1]{%
```

```

8469 \edef\@xdyattributes{\@xdyattributes ^^J \string"#1\string"}%
8470 \expandafter\toks@\expandafter{\@xdylocref}%
8471 \edef\@xdylocref{\the\toks@ ^^J%
8472 (markup-locref
8473 :open \string"\string~n\string\setentrycounter
8474 {\noexpand\glscounter}%
8475 \expandafter\string\csname#1\endcsname
8476 \expandafter\@gobble\string\{\string" ^^J
8477 :close \string"\expandafter\@gobble\string\}\string" ^^J
8478 :attr \string"#1\string")}}

```

Only has an effect before \writeist:

```

8479 \fi

```

\GlsAddXdyCounters

```

8480 \renewcommand*\GlsAddXdyCounters[1]{%
8481 \GlossariesWarning{\string\GlsAddXdyCounters\space not available
8482 in compatibility mode.}%
8483 }

```

Add predefined attributes

```

8484 \GlsAddXdyAttribute{glsnumberformat}
8485 \GlsAddXdyAttribute{textrm}
8486 \GlsAddXdyAttribute{textsf}
8487 \GlsAddXdyAttribute{texttt}
8488 \GlsAddXdyAttribute{textbf}
8489 \GlsAddXdyAttribute{textmd}
8490 \GlsAddXdyAttribute{textit}
8491 \GlsAddXdyAttribute{textup}
8492 \GlsAddXdyAttribute{textsl}
8493 \GlsAddXdyAttribute{textsc}
8494 \GlsAddXdyAttribute{emph}
8495 \GlsAddXdyAttribute{glshypernumber}
8496 \GlsAddXdyAttribute{hyperrrm}
8497 \GlsAddXdyAttribute{hypersf}
8498 \GlsAddXdyAttribute{hypertt}
8499 \GlsAddXdyAttribute{hyperbf}
8500 \GlsAddXdyAttribute{hypermd}
8501 \GlsAddXdyAttribute{hyperit}
8502 \GlsAddXdyAttribute{hyperup}
8503 \GlsAddXdyAttribute{hypersl}
8504 \GlsAddXdyAttribute{hypersc}
8505 \GlsAddXdyAttribute{hyperemph}

```

\GlsAddXdyLocation Restore v2.07 definition:

```

8506 \ifglxindy
8507 \renewcommand*\GlsAddXdyLocation[2]{%
8508 \edef\@xdyuserlocationdefs{%
8509 \@xdyuserlocationdefs ^^J%

```



```

8510      (define-location-class \string"#1\string"^^J\space\space
8511      \space(#2))
8512    }%
8513    \edef\xdyuserlocationnames{%
8514      \xdyuserlocationnames^^J\space\space\space
8515      \string"#1\string"}%
8516  }
8517\fi

```

\@do@wrglossary

```

8518 \renewcommand{\@do@wrglossary}[1]{%

```

Determine whether to use xindy or makeindex syntax

```

8519 \ifglxindy

```

Need to determine if the formatting information starts with a (or) indicating a range.

```

8520 \expandafter\@glo@check@mkidxrangechar\@glsnumberformat\@nil
8521 \def\@glo@range{%
8522 \expandafter\if\@glo@prefix(\relax
8523 \def\@glo@range{:open-range}%
8524 \else
8525 \expandafter\if\@glo@prefix)\relax
8526 \def\@glo@range{:close-range}%
8527 \fi
8528 \fi

```

Get the location and escape any special characters

```

8529 \protected@edef\@glslocref{\theglentrycounter}%
8530 \@gls@checkmkidxchars\@glslocref

```

Write to the glossary file using xindy syntax.

```

8531 \glossary[\csname glo@#1@type\endcsname]{%
8532 (indexentry :key (\csname glo@#1@index\endcsname)
8533 :locref \string"\@glslocref\string" %
8534 :attr \string"\@glo@suffix\string" \@glo@range
8535 )
8536 }%
8537 \else

```

Convert the format information into the format required for makeindex

```

8538 \@set@glo@numformat\@glo@numfmt\@gls@counter\@glsnumberformat

```

Write to the glossary file using makeindex syntax.

```

8539 \glossary[\csname glo@#1@type\endcsname]{%
8540 \string\glossaryentry{\csname glo@#1@index\endcsname
8541 \@gls@encapchar\@glo@numfmt}\theglentrycounter}}%
8542 \fi
8543 }

```

\@set@glo@numformat Only had 3 arguments in v2.07

```

8544 \def\@set@glo@numformat#1#2#3{%
8545   \expandafter\@glo@check@mkidxrangechar#3\@nil
8546   \protected@edef#1{%
8547     \@glo@prefix setentrycounter[]{#2}%
8548     \expandafter\string\csname\@glo@suffix\endcsname
8549   }%
8550   \@gls@checkmkidxchars#1%
8551 }

```

\writeist Redefine \writeist back to the way it was in v2.07, but change \istfile to
\glswrite.

```

8552 \ifglxindy
8553   \def\writeist{%
8554     \openout\glswrite=\istfilename
8555     \write\glswrite{;; xindy style file created by the glossaries
8556       package in compatible-2.07 mode}%
8557     \write\glswrite{;; for document '\jobname' on
8558       \the\year-\the\month-\the\day}%
8559     \write\glswrite{^^J; required styles^^J}
8560     \@for\@xdystyle:=\@xdyrequiredstyles\do{%
8561       \ifx\@xdystyle\@empty
8562         \else
8563           \protected@write\glswrite{{(require
8564             \string"\@xdystyle.xdy\string")}}%
8565         \fi
8566       }%
8567     \write\glswrite{^^J%
8568       ; list of allowed attributes (number formats)^^J}%
8569     \write\glswrite{(define-attributes ((\@xdyattributes)))}%
8570     \write\glswrite{^^J; user defined alphabets^^J}%
8571     \write\glswrite{\@xdyuseralphabets}%
8572     \write\glswrite{^^J; location class definitions^^J}%
8573     \protected@edef\@gls@roman{\@roman{0}\string"
8574       \string"roman-numbers-lowercase\string" :sep \string"}}%
8575     \@onelevel@sanitize\@gls@roman
8576     \edef\@tmp{\string" \string"roman-numbers-lowercase\string"
8577       :sep \string"}%
8578     \@onelevel@sanitize\@tmp
8579     \ifx\@tmp\@gls@roman
8580       \write\glswrite{(define-location-class
8581         \string"roman-page-numbers\string"^^J\space\space\space
8582         (\string"roman-numbers-lowercase\string")
8583         :min-range-length \@glsminrange)}}%
8584     \else
8585       \write\glswrite{(define-location-class
8586         \string"roman-page-numbers\string"^^J\space\space\space
8587         (:sep "\@gls@roman")
8588         :min-range-length \@glsminrange)}}%
8589     \fi

```

```

8590 \write\glswrite{(define-location-class
8591   \string"Roman-page-numbers\string"^^J\space\space\space
8592   (\string"roman-numbers-uppercase\string")
8593   :min-range-length \@glxminrange)}}%
8594 \write\glswrite{(define-location-class
8595   \string"arabic-page-numbers\string"^^J\space\space\space
8596   (\string"arabic-numbers\string")
8597   :min-range-length \@glxminrange)}}%
8598 \write\glswrite{(define-location-class
8599   \string"alpha-page-numbers\string"^^J\space\space\space
8600   (\string"alpha\string")
8601   :min-range-length \@glxminrange)}}%
8602 \write\glswrite{(define-location-class
8603   \string"Alpha-page-numbers\string"^^J\space\space\space
8604   (\string"ALPHA\string")
8605   :min-range-length \@glxminrange)}}%
8606 \write\glswrite{(define-location-class
8607   \string"Appendix-page-numbers\string"^^J\space\space\space
8608   (\string"ALPHA\string"
8609   :sep \string"\@glAlphacompositor\string"
8610   \string"arabic-numbers\string")
8611   :min-range-length \@glxminrange)}}%
8612 \write\glswrite{(define-location-class
8613   \string"arabic-section-numbers\string"^^J\space\space\space
8614   (\string"arabic-numbers\string"
8615   :sep \string"\glsc compositor\string"
8616   \string"arabic-numbers\string")
8617   :min-range-length \@glxminrange)}}%
8618 \write\glswrite{^^J; user defined location classes}%
8619 \write\glswrite{@xdyuserlocationdefs}%
8620 \write\glswrite{^^J; define cross-reference class^^J}%
8621 \write\glswrite{(define-crossref-class \string"see\string"
8622   :unverified )}%
8623 \write\glswrite{(markup-crossref-list
8624   :class \string"see\string"^^J\space\space\space
8625   :open \string"\string\glseeformat\string"
8626   :close \string"{}\string")}%
8627 \write\glswrite{^^J; define the order of the location classes}%
8628 \write\glswrite{(define-location-class-order
8629   (\@xdylocationclassorder))}%
8630 \write\glswrite{^^J; define the glossary markup^^J}%
8631 \write\glswrite{(markup-index^^J\space\space\space
8632   :open \string"\string
8633   \glossarysection[\string\glossarytoctitle]{\string
8634   \glossarytitle}\string\glossarypreamble\string~n\string\begin
8635   {theglossary}\string\glossaryheader\string~n\string" ^^J\space
8636   \space\space:close \string"\expandafter\@gobble
8637   \string\%\string~n\string
8638   \end{theglossary}\string\glossarypostamble

```

```

8639         \string~n\string" ^^J\space\space\space
8640         :tree)}}%
8641     \write\glswrite{(markup-letter-group-list
8642         :sep \string"\string\glsgroupskip\string~n\string"))}%
8643     \write\glswrite{(markup-indexentry
8644         :open \string"\string\relax \string\glsresetentrylist
8645         \string~n\string"))}%
8646     \write\glswrite{(markup-locclass-list :open
8647         \string"\glsopenbrace\string\glossaryentrynumbers
8648         \glsopenbrace\string\relax\space \string"^^J\space\space\space
8649         :sep \string", \string"
8650         :close \string"\glsclosebrace\glsclosebrace\string"))}%
8651     \write\glswrite{(markup-locref-list
8652         :sep \string"\string\delimN\space\string"))}%
8653     \write\glswrite{(markup-range
8654         :sep \string"\string\delimR\space\string"))}%
8655     \@onelevel@sanitize\gls@suffixF
8656     \@onelevel@sanitize\gls@suffixFF
8657     \ifx\gls@suffixF\@empty
8658     \else
8659         \write\glswrite{(markup-range
8660             :close "\gls@suffixF" :length 1 :ignore-end)}}%
8661     \fi
8662     \ifx\gls@suffixFF\@empty
8663     \else
8664         \write\glswrite{(markup-range
8665             :close "\gls@suffixFF" :length 2 :ignore-end)}}%
8666     \fi
8667     \write\glswrite{^^J; define format to use for locations^^J}%
8668     \write\glswrite{@xdylocref}%
8669     \write\glswrite{^^J; define letter group list format^^J}%
8670     \write\glswrite{(markup-letter-group-list
8671         :sep \string"\string\glsgroupskip\string~n\string"))}%
8672     \write\glswrite{^^J; letter group headings^^J}%
8673     \write\glswrite{(markup-letter-group
8674         :open-head \string"\string\glsgroupheading
8675         \glsopenbrace\string"^^J\space\space\space
8676         :close-head \string"\glsclosebrace\string"))}%
8677     \write\glswrite{^^J; additional letter groups^^J}%
8678     \write\glswrite{@xdylettergroups}%
8679     \write\glswrite{^^J; additional sort rules^^J}%
8680     \write\glswrite{@xdysortrules}%
8681     \noist}
8682 \else
8683     \edef\@gls@actualchar{\string?}
8684     \edef\@gls@encapchar{\string|}
8685     \edef\@gls@levelchar{\string!}
8686     \edef\@gls@quotechar{\string"}
8687     \def\writeist{\relax

```

```

8688 \openout\glswrite=\istfilename
8689 \write\glswrite{\expandafter\@gobble\string\% makeindex style file
8690   created by the glossaries package}
8691 \write\glswrite{\expandafter\@gobble\string\% for document
8692   '\jobname' on \the\year-\the\month-\the\day}
8693 \write\glswrite{actual '\@gls@actualchar'}
8694 \write\glswrite{encap '\@gls@encapchar'}
8695 \write\glswrite{level '\@gls@levelchar'}
8696 \write\glswrite{quote '\@gls@quotechar'}
8697 \write\glswrite{keyword \string"\string\glossaryentry\string"}
8698 \write\glswrite{preamble \string"\string\glossarysection[\string
8699   \glossarytoctitle]{\string\glossarytitle}\string
8700   \glossarypreamble\string\n\string\begin{theglossary}\string
8701   \glossaryheader\string\n\string"}
8702 \write\glswrite{postamble \string"\string\% \string\n\string
8703   \end{theglossary}\string\glossarypostamble\string\n
8704   \string"}
8705 \write\glswrite{group_skip \string"\string\glsgroupskip\string\n
8706   \string"}
8707 \write\glswrite{item_0 \string"\string\% \string\n\string"}
8708 \write\glswrite{item_1 \string"\string\% \string\n\string"}
8709 \write\glswrite{item_2 \string"\string\% \string\n\string"}
8710 \write\glswrite{item_01 \string"\string\% \string\n\string"}
8711 \write\glswrite{item_x1
8712   \string"\string\relax \string\glsresetentrylist\string\n
8713   \string"}
8714 \write\glswrite{item_12 \string"\string\% \string\n\string"}
8715 \write\glswrite{item_x2
8716   \string"\string\relax \string\glsresetentrylist\string\n
8717   \string"}
8718 \write\glswrite{delim_0 \string"\string\{\string
8719   \glossaryentrynumbers\string\{\string\relax \string"}
8720 \write\glswrite{delim_1 \string"\string\{\string
8721   \glossaryentrynumbers\string\{\string\relax \string"}
8722 \write\glswrite{delim_2 \string"\string\{\string
8723   \glossaryentrynumbers\string\{\string\relax \string"}
8724 \write\glswrite{delim_t \string"\string\}\string\}\string"}
8725 \write\glswrite{delim_n \string"\string\delimN \string"}
8726 \write\glswrite{delim_r \string"\string\delimR \string"}
8727 \write\glswrite{headings_flag 1}
8728 \write\glswrite{heading_prefix
8729   \string"\string\glsgroupheading\string\{\string"}
8730 \write\glswrite{heading_suffix
8731   \string"\string\}\string\relax
8732   \string\glsresetentrylist \string"}
8733 \write\glswrite{symhead_positive \string"glssymbols\string"}
8734 \write\glswrite{numhead_positive \string"glslnumbers\string"}
8735 \write\glswrite{page_compositor \string"glscpositor\string"}
8736 \@gls@escbsdq\gls@suffixF

```

```

8737 \@gls@escbsdq\gls@suffixFF
8738 \ifx\gls@suffixF\@empty
8739 \else
8740 \write\glswrite{suffix_2p \string"\gls@suffixF\string"}
8741 \fi
8742 \ifx\gls@suffixFF\@empty
8743 \else
8744 \write\glswrite{suffix_3p \string"\gls@suffixFF\string"}
8745 \fi
8746 \noist
8747 }
8748 \fi

```

\noist

```

8749 \renewcommand*{\noist}{\let\writeist\relax}

```

Compatibility macros.

```

8750 \NeedsTeXFormat{LaTeX2e}
8751 \ProvidesPackage{glossaries-compatible-307}[2013/11/14 v4.0 (NLCT)]

```

Compatibility macros for predefined glossary styles:

compatglossarystyle Defines a compatibility glossary style.

```

8752 \newcommand{\compatglossarystyle}[2]{%
8753 \ifcsundef{@glscompstyle@#1}%
8754 {%
8755 \csdef{@glscompstyle@#1}{#2}%
8756 }%
8757 {%
8758 \PackageError{glossaries}{Glossary compatibility style ‘#1’ is already defined}{}%
8759 }%
8760 }

```

Backward compatible inline style.

```

8761 \compatglossarystyle{inline}{%
8762 \renewcommand{\glossaryentryfield}[5]{%
8763 \glsinlinedopostchild
8764 \gls@inlinesep
8765 \def\glo@desc{##3}%
8766 \def\@no@post@desc{\nopostdesc}%
8767 \glsentryitem{##1}\glsinlinenameformat{##1}{##2}%
8768 \ifx\glo@desc\@no@post@desc
8769 \glsinlineemptydescformat{##4}{##5}%
8770 \else
8771 \ifstrempy{##3}%
8772 {\glsinlineemptydescformat{##4}{##5}}%
8773 {\glsinlinedescformat{##3}{##4}{##5}}%
8774 \fi
8775 \ifglshaschildren{##1}%
8776 {%

```

```

8777 \glsresetsubentrycounter
8778 \glsinlineparentchildseparator
8779 \def\gls@inlinesubsep{%
8780 \def\gls@inlinepostchild{\glsinlinepostchild}%
8781 }%
8782 {}%
8783 \def\gls@inlinesep{\glsinlineseparator}%
8784 }%

```

Sub-entries display description:

```

8785 \renewcommand{\glossarysubentryfield}[6]{%
8786 \gls@inlinesubsep%
8787 \glsinlinesubnameformat{##2}{##3}%
8788 \glssubentryitem{##2}\glsinlinesubdescformat{##4}{##5}{##6}%
8789 \def\gls@inlinesubsep{\glsinlinesubseparator}%
8790 }%
8791 }

```

Backward compatible list style.

```

8792 \compatglossarystyle{list}{%
8793 \renewcommand*\glossaryentryfield}[5]{%
8794 \item[\glsentryitem{##1}\glstarget{##1}{##2}]
8795 ##3\glspostdescription\space ##5}%

```

Sub-entries continue on the same line:

```

8796 \renewcommand*\glossarysubentryfield}[6]{%
8797 \glssubentryitem{##2}%
8798 \glstarget{##2}{\strut}##4\glspostdescription\space ##6.}%
8799 }

```

Backward compatible listgroup style.

```

8800 \compatglossarystyle{listgroup}{%
8801 \csuse{@glscompstyle@list}%
8802 }%

```

Backward compatible listhypergroup style.

```

8803 \compatglossarystyle{listhypergroup}{%
8804 \csuse{@glscompstyle@list}%
8805 }%

```

Backward compatible altlist style.

```

8806 \compatglossarystyle{altlist}{%
8807 \renewcommand*\glossaryentryfield}[5]{%
8808 \item[\glsentryitem{##1}\glstarget{##1}{##2}]%
8809 \mbox{}\par\nobreak\@afterheading
8810 ##3\glspostdescription\space ##5}%
8811 \renewcommand{\glossarysubentryfield}[6]{%
8812 \par
8813 \glssubentryitem{##2}%
8814 \glstarget{##2}{\strut}##4\glspostdescription\space ##6}%
8815 }%

```

Backward compatible altlistgroup style.

```
8816 \compatglossarystyle{altlistgroup}{%
8817 \csuse{@glscompstyle@altlist}%
8818 }%
```

Backward compatible altlisthypergroup style.

```
8819 \compatglossarystyle{altlisthypergroup}{%
8820 \csuse{@glscompstyle@altlist}%
8821 }%
```

Backward compatible listdotted style.

```
8822 \compatglossarystyle{listdotted}{%
8823 \renewcommand*{\glossaryentryfield}[5]{%
8824 \item[]\makebox[\glslistdottedwidth][l]{%
8825 \glentryitem{##1}\glstarget{##1}{##2}%
8826 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##3}%
8827 \renewcommand*{\glossarysubentryfield}[6]{%
8828 \item[]\makebox[\glslistdottedwidth][l]{%
8829 \glssubentryitem{##2}%
8830 \glstarget{##2}{##3}%
8831 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##4}%
8832 }%
```

Backward compatible sublistdotted style.

```
8833 \compatglossarystyle{sublistdotted}{%
8834 \csuse{@glscompstyle@listdotted}%
8835 \renewcommand*{\glossaryentryfield}[5]{%
8836 \item[\glentryitem{##1}\glstarget{##1}{##2}]}%
8837 }%
```

Backward compatible long style.

```
8838 \compatglossarystyle{long}{%
8839 \renewcommand*{\glossaryentryfield}[5]{%
8840 \glentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\}%
8841 \renewcommand*{\glossarysubentryfield}[6]{%
8842 &
8843 \glssubentryitem{##2}%
8844 \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\}%
8845 }%
```

Backward compatible longborder style.

```
8846 \compatglossarystyle{longborder}{%
8847 \csuse{@glscompstyle@long}%
8848 }%
```

Backward compatible longheader style.

```
8849 \compatglossarystyle{longheader}{%
8850 \csuse{@glscompstyle@long}%
8851 }%
```

Backward compatible longheaderborder style.

```
8852 \compatglossarystyle{longheaderborder}{%
```



```

8853 \csuse{@glscompstyle@long}%
8854 }%

```

Backward compatible long3col style.

```

8855 \compatglossarystyle{long3col}{%
8856   \renewcommand*{\glossaryentryfield}[5]{%
8857     \glsentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\\}%
8858   \renewcommand*{\glossarysubentryfield}[6]{%
8859     &
8860     \glssubentryitem{##2}%
8861     \glstarget{##2}{\strut}##4 & ##6\\}%
8862 }%

```

Backward compatible long3colborder style.

```

8863 \compatglossarystyle{long3colborder}{%
8864   \csuse{@glscompstyle@long3col}%
8865 }%

```

Backward compatible long3colheader style.

```

8866 \compatglossarystyle{long3colheader}{%
8867   \csuse{@glscompstyle@long3col}%
8868 }%

```

Backward compatible long3colheaderborder style.

```

8869 \compatglossarystyle{long3colheaderborder}{%
8870   \csuse{@glscompstyle@long3col}%
8871 }%

```

Backward compatible long4col style.

```

8872 \compatglossarystyle{long4col}{%
8873   \renewcommand*{\glossaryentryfield}[5]{%
8874     \glsentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\\}%
8875   \renewcommand*{\glossarysubentryfield}[6]{%
8876     &
8877     \glssubentryitem{##2}%
8878     \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%
8879 }%

```

Backward compatible long4colheader style.

```

8880 \compatglossarystyle{long4colheader}{%
8881   \csuse{@glscompstyle@long4col}%
8882 }%

```

Backward compatible long4colborder style.

```

8883 \compatglossarystyle{long4colborder}{%
8884   \csuse{@glscompstyle@long4col}%
8885 }%

```

Backward compatible long4colheaderborder style.

```

8886 \compatglossarystyle{long4colheaderborder}{%
8887   \csuse{@glscompstyle@long4col}%
8888 }%

```

Backward compatible altlong4col style.

```
8889 \compatglossarystyle{altlong4col}{%
8890 \csuse{@glscompstyle@long4col}%
8891 }%
```

Backward compatible altlong4colheader style.

```
8892 \compatglossarystyle{altlong4colheader}{%
8893 \csuse{@glscompstyle@long4col}%
8894 }%
```

Backward compatible altlong4colborder style.

```
8895 \compatglossarystyle{altlong4colborder}{%
8896 \csuse{@glscompstyle@long4col}%
8897 }%
```

Backward compatible altlong4colheaderborder style.

```
8898 \compatglossarystyle{altlong4colheaderborder}{%
8899 \csuse{@glscompstyle@long4col}%
8900 }%
```

Backward compatible long style.

```
8901 \compatglossarystyle{longragged}{%
8902 \renewcommand*{\glossaryentryfield}[5]{%
8903 \glstryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%
8904 \tabularnewline}%
8905 \renewcommand*{\glossarysubentryfield}[6]{%
8906 &
8907 \glssubentryitem{##2}%
8908 \glstarget{##2}{\strut}##4\glspostdescription\space ##6%
8909 \tabularnewline}%
8910 }%
```

Backward compatible longraggedborder style.

```
8911 \compatglossarystyle{longraggedborder}{%
8912 \csuse{@glscompstyle@longragged}%
8913 }%
```

Backward compatible longraggedheader style.

```
8914 \compatglossarystyle{longraggedheader}{%
8915 \csuse{@glscompstyle@longragged}%
8916 }%
```

Backward compatible longraggedheaderborder style.

```
8917 \compatglossarystyle{longraggedheaderborder}{%
8918 \csuse{@glscompstyle@longragged}%
8919 }%
```

Backward compatible longragged3col style.

```
8920 \compatglossarystyle{longragged3col}{%
8921 \renewcommand*{\glossaryentryfield}[5]{%
8922 \glstryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\tabularnewline}%
8923 \renewcommand*{\glossarysubentryfield}[6]{%
```

```

8924      &
8925      \glssubentryitem{##2}%
8926      \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%
8927 }%

Backward compatible longragged3colborder style.
8928 \compatglossarystyle{longragged3colborder}{%
8929 \csuse{@glscmpstyle@longragged3col}%
8930 }%

Backward compatible longragged3colheader style.
8931 \compatglossarystyle{longragged3colheader}{%
8932 \csuse{@glscmpstyle@longragged3col}%
8933 }%

Backward compatible longragged3colheaderborder style.
8934 \compatglossarystyle{longragged3colheaderborder}{%
8935 \csuse{@glscmpstyle@longragged3col}%
8936 }%

Backward compatible altlongragged4col style.
8937 \compatglossarystyle{altlongragged4col}{%
8938 \renewcommand*{\glossaryentryfield}[5]{%
8939 \glssentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\tabularnewline}%
8940 \renewcommand*{\glossarysubentryfield}[6]{%
8941 &
8942 \glssubentryitem{##2}%
8943 \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%
8944 }%

Backward compatible altlongragged4colheader style.
8945 \compatglossarystyle{altlongragged4colheader}{%
8946 \csuse{@glscmpstyle@altlong4col}%
8947 }%

Backward compatible altlongragged4colborder style.
8948 \compatglossarystyle{altlongragged4colborder}{%
8949 \csuse{@glscmpstyle@altlong4col}%
8950 }%

Backward compatible altlongragged4colheaderborder style.
8951 \compatglossarystyle{altlongragged4colheaderborder}{%
8952 \csuse{@glscmpstyle@altlong4col}%
8953 }%

Backward compatible index style.
8954 \compatglossarystyle{index}{%
8955 \renewcommand*{\glossaryentryfield}[5]{%
8956 \item\glssentryitem{##1}\textbf{\glstarget{##1}{##2}}%
8957 \ifx\relax##4\relax
8958 \else
8959 \space{##4}%

```

```

8960      \fi
8961      \space ##3\glspostdescription \space ##5}%
8962 \renewcommand*{\glossarysubentryfield}[6]{%
8963   \ifcase##1\relax
8964     % level 0
8965     \item
8966   \or
8967     % level 1
8968     \subitem
8969     \glssubentryitem{##2}%
8970   \else
8971     % all other levels
8972     \subsubitem
8973   \fi
8974   \textbf{\glstarget{##2}{##3}}%
8975   \ifx\relax##5\relax
8976   \else
8977     \space(##5)%
8978   \fi
8979   \space##4\glspostdescription\space ##6}%
8980 }%

```

Backward compatible indexgroup style.

```

8981 \compatglossarystyle{indexgroup}{%
8982  \csuse{@glscmpstyle@index}%
8983 }%

```

Backward compatible indexhypergroup style.

```

8984 \compatglossarystyle{indexhypergroup}{%
8985  \csuse{@glscmpstyle@index}%
8986 }%

```

Backward compatible tree style.

```

8987 \compatglossarystyle{tree}{%
8988  \renewcommand{\glossaryentryfield}[5]{%
8989    \hangindent0pt\relax
8990    \parindent0pt\relax
8991    \glstentryitem{##1}\textbf{\glstarget{##1}{##2}}%
8992    \ifx\relax##4\relax
8993    \else
8994      \space(##4)%
8995    \fi
8996    \space ##3\glspostdescription \space ##5\par}%
8997 \renewcommand{\glossarysubentryfield}[6]{%
8998   \hangindent##1\glstreeindent\relax
8999   \parindent##1\glstreeindent\relax
9000   \ifnum##1=1\relax
9001     \glssubentryitem{##2}%
9002   \fi
9003   \textbf{\glstarget{##2}{##3}}%
9004   \ifx\relax##5\relax

```

```

9005     \else
9006     \space(##5)%
9007     \fi
9008     \space##4\glspostdescription\space ##6\par}%
9009 }%

    Backward compatible treegroup style.
9010 \compatglossarystyle{treegroup}{%
9011 \csuse{@glscmpstyle@tree}%
9012 }%

    Backward compatible treehypergroup style.
9013 \compatglossarystyle{treehypergroup}{%
9014 \csuse{@glscmpstyle@tree}%
9015 }%

    Backward compatible treenoname style.
9016 \compatglossarystyle{treenoname}{%
9017 \renewcommand{\glossaryentryfield}[5]{%
9018 \hangindent0pt\relax
9019 \parindent0pt\relax
9020 \glstentryitem{##1}\textbf{\glstarget{##1}{##2}}%
9021 \ifx\relax##4\relax
9022 \else
9023 \space(##4)%
9024 \fi
9025 \space ##3\glspostdescription \space ##5\par}%
9026 \renewcommand{\glossarysubentryfield}[6]{%
9027 \hangindent##1\glstreeindent\relax
9028 \parindent##1\glstreeindent\relax
9029 \ifnum##1=1\relax
9030 \glssubentryitem{##2}%
9031 \fi
9032 \glstarget{##2}{\strut}%
9033 ##4\glspostdescription\space ##6\par}%
9034 }%

    Backward compatible treenonamegroup style.
9035 \compatglossarystyle{treenonamegroup}{%
9036 \csuse{@glscmpstyle@treenoname}%
9037 }%

    Backward compatible treenonamehypergroup style.
9038 \compatglossarystyle{treenonamehypergroup}{%
9039 \csuse{@glscmpstyle@treenoname}%
9040 }%

    Backward compatible alttree style.
9041 \compatglossarystyle{alttree}{%
9042 \renewcommand{\glossaryentryfield}[5]{%
9043 \ifnum\@gls@prevlevel=0\relax
9044 \else

```

```

9045     \settowidth{\glstreeindent}{\textbf{\@glswidestname\space}}%
9046     \hangindent\glstreeindent
9047     \parindent\glstreeindent
9048     \fi
9049     \makebox[0pt][r]{\makebox[\glstreeindent][l]{%
9050       \glstentryitem{##1}\textbf{\glstarget{##1}{##2}}}%
9051     \ifx\relax##4\relax
9052     \else
9053       (##4)\space
9054     \fi
9055     ##3\glspostdescription \space ##5\par
9056     \def\@gls@prevlevel{0}%
9057   }%
9058   \renewcommand{\glossarysubentryfield}[6]{%
9059     \ifnum##1=1\relax
9060       \glssubentryitem{##2}%
9061     \fi
9062     \ifnum\@gls@prevlevel=##1\relax
9063     \else
9064       \@ifundefined{\@glswidestname\romannumeral##1}{%
9065         \settowidth{\gls@tmplen}{\textbf{\@glswidestname\space}}{%
9066         \settowidth{\gls@tmplen}{\textbf{%
9067           \csname \@glswidestname\romannumeral##1\endcsname\space}}}%
9068       \ifnum\@gls@prevlevel<##1\relax
9069         \setlength\glstreeindent\gls@tmplen
9070         \addtolength\glstreeindent\parindent
9071         \parindent\glstreeindent
9072       \else
9073         \@ifundefined{\@glswidestname\romannumeral\@gls@prevlevel}{%
9074           \settowidth{\glstreeindent}{\textbf{%
9075             \@glswidestname\space}}{%
9076           \settowidth{\glstreeindent}{\textbf{%
9077             \csname \@glswidestname\romannumeral\@gls@prevlevel
9078               \endcsname\space}}}%
9079         \addtolength\parindent{-\glstreeindent}%
9080         \setlength\glstreeindent\parindent
9081       \fi
9082     \fi
9083     \hangindent\glstreeindent
9084     \makebox[0pt][r]{\makebox[\gls@tmplen][l]{%
9085       \textbf{\glstarget{##2}{##3}}}%
9086     \ifx##5\relax\relax
9087     \else
9088       (##5)\space
9089     \fi
9090     ##4\glspostdescription\space ##6\par
9091     \def\@gls@prevlevel{##1}%
9092   }%
9093 }%

```

Backward compatible alttreegroup style.

```
9094 \compatglossarystyle{alttreegroup}{%  
9095 \csuse{@glscompstyle@alttree}%  
9096 }%
```

Backward compatible alttreehypergroup style.

```
9097 \compatglossarystyle{alttreehypergroup}{%  
9098 \csuse{@glscompstyle@alttree}%  
9099 }%
```

Backward compatible mcolindex style.

```
9100 \compatglossarystyle{mcolindex}{%  
9101 \csuse{@glscompstyle@index}%  
9102 }%
```

Backward compatible mcolindexgroup style.

```
9103 \compatglossarystyle{mcolindexgroup}{%  
9104 \csuse{@glscompstyle@index}%  
9105 }%
```

Backward compatible mcolindexhypergroup style.

```
9106 \compatglossarystyle{mcolindexhypergroup}{%  
9107 \csuse{@glscompstyle@index}%  
9108 }%
```

Backward compatible mcoltree style.

```
9109 \compatglossarystyle{mcoltree}{%  
9110 \csuse{@glscompstyle@tree}%  
9111 }%
```

Backward compatible mcoltreegroup style.

```
9112 \compatglossarystyle{mcolindextreegroup}{%  
9113 \csuse{@glscompstyle@tree}%  
9114 }%
```

Backward compatible mcoltreehypergroup style.

```
9115 \compatglossarystyle{mcolindextreehypergroup}{%  
9116 \csuse{@glscompstyle@tree}%  
9117 }%
```

Backward compatible mcoltreenoname style.

```
9118 \compatglossarystyle{mcoltreenoname}{%  
9119 \csuse{@glscompstyle@tree}%  
9120 }%
```

Backward compatible mcoltreenonamegroup style.

```
9121 \compatglossarystyle{mcoltreenonamegroup}{%  
9122 \csuse{@glscompstyle@tree}%  
9123 }%
```

Backward compatible mcoltreenonamehypergroup style.

```
9124 \compatglossarystyle{mcoltreenonamehypergroup}{%  
9125 \csuse{@glscompstyle@tree}%  
9126 }%
```

Backward compatible mcolalmtree style.

```
9127 \compatglossarystyle{mcolalmtree}{%
9128 \csuse{@glscmpstyle@almtree}%
9129 }%
```

Backward compatible mcolalmtreegroup style.

```
9130 \compatglossarystyle{mcolalmtreegroup}{%
9131 \csuse{@glscmpstyle@almtree}%
9132 }%
```

Backward compatible mcolalmtreehypergroup style.

```
9133 \compatglossarystyle{mcolalmtreehypergroup}{%
9134 \csuse{@glscmpstyle@almtree}%
9135 }%
```

Backward compatible superragged style.

```
9136 \compatglossarystyle{superragged}{%
9137 \renewcommand*{\glossaryentryfield}[5]{%
9138 \glstryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%
9139 \tabularnewline}%
9140 \renewcommand*{\glossarysubentryfield}[6]{%
9141 &
9142 \glssubentryitem{##2}%
9143 \glstarget{##2}{\strut}##4\glspostdescription\space ##6%
9144 \tabularnewline}%
9145 }%
```

Backward compatible superraggedborder style.

```
9146 \compatglossarystyle{superraggedborder}{%
9147 \csuse{@glscmpstyle@superragged}%
9148 }%
```

Backward compatible superraggedheader style.

```
9149 \compatglossarystyle{superraggedheader}{%
9150 \csuse{@glscmpstyle@superragged}%
9151 }%
```

Backward compatible superraggedheaderborder style.

```
9152 \compatglossarystyle{superraggedheaderborder}{%
9153 \csuse{@glscmpstyle@superragged}%
9154 }%
```

Backward compatible superragged3col style.

```
9155 \compatglossarystyle{superragged3col}{%
9156 \renewcommand*{\glossaryentryfield}[5]{%
9157 \glstryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\tabularnewline}%
9158 \renewcommand*{\glossarysubentryfield}[6]{%
9159 &
9160 \glssubentryitem{##2}%
9161 \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%
9162 }%
```


Backward compatible superragged3colborder style.

```
9163 \compatglossarystyle{superragged3colborder}{%
9164 \csuse{@glscmpstyle@superragged3col}%
9165 }%
```

Backward compatible superragged3colheader style.

```
9166 \compatglossarystyle{superragged3colheader}{%
9167 \csuse{@glscmpstyle@superragged3col}%
9168 }%
```

Backward compatible superragged3colheaderborder style.

```
9169 \compatglossarystyle{superragged3colheaderborder}{%
9170 \csuse{@glscmpstyle@superragged3col}%
9171 }%
```

Backward compatible altsuperragged4col style.

```
9172 \compatglossarystyle{altsuperragged4col}{%
9173 \renewcommand*{\glossaryentryfield}[5]{%
9174 \glstarget{##1}{\glstarget{##1}{##2} & ##3 & ##4 & ##5\tabularnewline}%
9175 \renewcommand*{\glossarysubentryfield}[6]{%
9176 &
9177 \glssubentryitem{##2}%
9178 \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%
9179 }%
```

Backward compatible altsuperragged4colheader style.

```
9180 \compatglossarystyle{altsuperragged4colheader}{%
9181 \csuse{@glscmpstyle@altsuperragged4col}%
9182 }%
```

Backward compatible altsuperragged4colborder style.

```
9183 \compatglossarystyle{altsuperragged4colborder}{%
9184 \csuse{@glscmpstyle@altsuperragged4col}%
9185 }%
```

Backward compatible altsuperragged4colheaderborder style.

```
9186 \compatglossarystyle{altsuperragged4colheaderborder}{%
9187 \csuse{@glscmpstyle@altsuperragged4col}%
9188 }%
```

Backward compatible super style.

```
9189 \compatglossarystyle{super}{%
9190 \renewcommand*{\glossaryentryfield}[5]{%
9191 \glstarget{##1}{\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\}%
9192 \renewcommand*{\glossarysubentryfield}[6]{%
9193 &
9194 \glssubentryitem{##2}%
9195 \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\}%
9196 }%
```

Backward compatible superborder style.

```
9197 \compatglossarystyle{superborder}{%
```

```

9198 \csuse{@glscompstyle@super}%
9199 }%

```

Backward compatible superheader style.

```

9200 \compatglossarystyle{superheader}{%
9201 \csuse{@glscompstyle@super}%
9202 }%

```

Backward compatible superheaderborder style.

```

9203 \compatglossarystyle{superheaderborder}{%
9204 \csuse{@glscompstyle@super}%
9205 }%

```

Backward compatible super3col style.

```

9206 \compatglossarystyle{super3col}{%
9207 \renewcommand*{\glossaryentryfield}[5]{%
9208 \glstarget{##1}{\glstarget{##1}{##2} & ##3 & ##5\\}%
9209 \renewcommand*{\glossarysubentryfield}[6]{%
9210 &
9211 \glssubentryitem{##2}%
9212 \glstarget{##2}{\strut}##4 & ##6\\}%
9213 }%

```

Backward compatible super3colborder style.

```

9214 \compatglossarystyle{super3colborder}{%
9215 \csuse{@glscompstyle@super3col}%
9216 }%

```

Backward compatible super3colheader style.

```

9217 \compatglossarystyle{super3colheader}{%
9218 \csuse{@glscompstyle@super3col}%
9219 }%

```

Backward compatible super3colheaderborder style.

```

9220 \compatglossarystyle{super3colheaderborder}{%
9221 \csuse{@glscompstyle@super3col}%
9222 }%

```

Backward compatible super4col style.

```

9223 \compatglossarystyle{super4col}{%
9224 \renewcommand*{\glossaryentryfield}[5]{%
9225 \glstarget{##1}{\glstarget{##1}{##2} & ##3 & ##4 & ##5\\}%
9226 \renewcommand*{\glossarysubentryfield}[6]{%
9227 &
9228 \glssubentryitem{##2}%
9229 \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%
9230 }%

```

Backward compatible super4colheader style.

```

9231 \compatglossarystyle{super4colheader}{%
9232 \csuse{@glscompstyle@super4col}%
9233 }%

```

Backward compatible super4colborder style.

```
9234 \compatglossarystyle{super4colborder}{%
9235 \csuse{@glscmpstyle@super4col}%
9236 }%
```

Backward compatible super4colheaderborder style.

```
9237 \compatglossarystyle{super4colheaderborder}{%
9238 \csuse{@glscmpstyle@super4col}%
9239 }%
```

Backward compatible altsuper4col style.

```
9240 \compatglossarystyle{altsuper4col}{%
9241 \csuse{@glscmpstyle@super4col}%
9242 }%
```

Backward compatible altsuper4colheader style.

```
9243 \compatglossarystyle{altsuper4colheader}{%
9244 \csuse{@glscmpstyle@super4col}%
9245 }%
```

Backward compatible altsuper4colborder style.

```
9246 \compatglossarystyle{altsuper4colborder}{%
9247 \csuse{@glscmpstyle@super4col}%
9248 }%
```

Backward compatible altsuper4colheaderborder style.

```
9249 \compatglossarystyle{altsuper4colheaderborder}{%
9250 \csuse{@glscmpstyle@super4col}%
9251 }%
```

7 Accessibility Support (glossaries-accsupp Code)

The package is experimental. It is intended to provide a means of using the PDF accessibility support in glossary entries. See the documentation for further details about accessibility support.

```
9252 \NeedsTeXFormat{LaTeX2e}
```

Package version number now in line with main glossaries package number but will only be updated when glossaries-accsupp.sty is modified.

```
9253 \ProvidesPackage{glossaries-accsupp}[2014/07/30 v4.08 (NLCT)]
```

```
9254 Experimental glossaries accessibility
```

Pass all options to glossaries:

```
9255 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{glossaries}}
```

Process options:

```
9256 \ProcessOptions
```

`\compatibleglossentry` Override style compatibility macros:

```
9257 \def\compatibleglossentry#1#2{%
```

```

9258 \toks@{#2}%
9259 \protected@edef\@do@glossentry{%
9260   \noexpand\accsuppglossaryentryfield{#1}%
9261   {\noexpand\glsnamefont
9262     {\expandafter\expandonce\csname glo@\glsdetoklabel{#1}@name\endcsname}}%
9263   {\expandafter\expandonce\csname glo@\glsdetoklabel{#1}@desc\endcsname}%
9264   {\expandafter\expandonce\csname glo@\glsdetoklabel{#1}@symbol\endcsname}%
9265   {\the\toks@}%
9266 }%
9267 \@do@glossentry
9268 }

```

atiblessubglossentry

```

9269 \def\compatiblessubglossentry#1#2#3{%
9270   \toks@{#3}%
9271   \protected@edef\@do@subglossentry{%
9272     \noexpand\accsuppglossarysubentryfield{\number#1}%
9273     {#2}%
9274     {\noexpand\glsnamefont
9275       {\expandafter\expandonce\csname glo@\glsdetoklabel{#2}@name\endcsname}}%
9276     {\expandafter\expandonce\csname glo@\glsdetoklabel{#2}@desc\endcsname}%
9277     {\expandafter\expandonce\csname glo@\glsdetoklabel{#2}@symbol\endcsname}%
9278     {\the\toks@}%
9279   }%
9280   \@do@subglossentry
9281 }

```

Required packages:

```

9282 \RequirePackage{glossaries}
9283 \RequirePackage{accsupp}

```

7.1 Defining Replacement Text

The version 0.1 stored the replacement text in the symbol key. This has been changed to use the new keys defined here. Example of use:

```
\newglossaryentry{dr}{name=Dr,description={},access={Doctor}}
```

access The replacement text corresponding to the name key:

```

9284 \define@key{glossentry}{access}{%
9285   \def\@glo@access{#1}%
9286 }

```

textaccess The replacement text corresponding to the text key:

```

9287 \define@key{glossentry}{textaccess}{%
9288   \def\@glo@textaccess{#1}%
9289 }

```

firstaccess The replacement text corresponding to the first key:

```
9290 \define@key{glossentry}{firstaccess}{%
9291   \def\@glo@firstaccess{#1}%
9292 }
```

pluralaccess The replacement text corresponding to the plural key:

```
9293 \define@key{glossentry}{pluralaccess}{%
9294   \def\@glo@pluralaccess{#1}%
9295 }
```

firstpluralaccess The replacement text corresponding to the firstplural key:

```
9296 \define@key{glossentry}{firstpluralaccess}{%
9297   \def\@glo@firstpluralaccess{#1}%
9298 }
```

symbolaccess The replacement text corresponding to the symbol key:

```
9299 \define@key{glossentry}{symbolaccess}{%
9300   \def\@glo@symbolaccess{#1}%
9301 }
```

symbolpluralaccess The replacement text corresponding to the symbolplural key:

```
9302 \define@key{glossentry}{symbolpluralaccess}{%
9303   \def\@glo@symbolpluralaccess{#1}%
9304 }
```

descriptionaccess The replacement text corresponding to the description key:

```
9305 \define@key{glossentry}{descriptionaccess}{%
9306   \def\@glo@descaccess{#1}%
9307 }
```

descriptionpluralaccess The replacement text corresponding to the descriptionplural key:

```
9308 \define@key{glossentry}{descriptionpluralaccess}{%
9309   \def\@glo@descpluralaccess{#1}%
9310 }
```

shortaccess The replacement text corresponding to the short key:

```
9311 \define@key{glossentry}{shortaccess}{%
9312   \def\@glo@shortaccess{#1}%
9313 }
```

shortpluralaccess The replacement text corresponding to the shortplural key:

```
9314 \define@key{glossentry}{shortpluralaccess}{%
9315   \def\@glo@shortpluralaccess{#1}%
9316 }
```

longaccess The replacement text corresponding to the long key:

```
9317 \define@key{glossentry}{longaccess}{%
9318   \def\@glo@longaccess{#1}%
9319 }
```

longpluralaccess The replacement text corresponding to the longplural key:

```
9320 \define@key{glossentry}{longpluralaccess}{%
9321   \def\@glo@longpluralaccess{#1}%
9322 }
```

There are no equivalent keys for the user1...user6 keys. The replacement text would have to be explicitly put in the value, e.g., user1={\glsacccsupp{inches}{in}}.

Append these new keys to \@gls@keymap:

```
9323 \appto\@gls@keymap{,%
9324   {access}{access},%
9325   {textaccess}{textaccess},%
9326   {firstaccess}{firstaccess},%
9327   {pluralaccess}{pluralaccess},%
9328   {firstpluralaccess}{firstpluralaccess},%
9329   {symbolaccess}{symbolaccess},%
9330   {symbolpluralaccess}{symbolpluralaccess},%
9331   {descaccess}{descaccess},%
9332   {descpluralaccess}{descpluralaccess},%
9333   {shortaccess}{shortaccess},%
9334   {shortpluralaccess}{shortpluralaccess},%
9335   {longaccess}{longaccess},%
9336   {longpluralaccess}{longpluralaccess}%
9337 }
```

\@gls@noaccess Indicates that no replacement text has been provided.

```
9338 \def\@gls@noaccess{\relax}
```

Add to the start hook (the access key is initialised to the value of the symbol key at the start for backwards compatibility):

```
9339 \let\@gls@oldnewglossaryentryprehook\@newglossaryentryprehook
9340 \renewcommand*{\@newglossaryentryprehook}{%
9341   \@gls@oldnewglossaryentryprehook
9342   \def\@glo@access{\@glo@symbol}%

```

Initialise the other keys:

```
9343   \def\@glo@textaccess{\@glo@access}%
9344   \def\@glo@firstaccess{\@glo@access}%
9345   \def\@glo@pluralaccess{\@glo@textaccess}%
9346   \def\@glo@firstpluralaccess{\@glo@pluralaccess}%
9347   \def\@glo@symbolaccess{\relax}%
9348   \def\@glo@symbolpluralaccess{\@glo@symbolaccess}%
9349   \def\@glo@descaccess{\relax}%
9350   \def\@glo@descpluralaccess{\@glo@descaccess}%
9351   \def\@glo@shortaccess{\relax}%
9352   \def\@glo@shortpluralaccess{\@glo@shortaccess}%
9353   \def\@glo@longaccess{\relax}%
9354   \def\@glo@longpluralaccess{\@glo@longaccess}%
9355 }
```

Add to the end hook:

```
9356 \let\@gls@oldnewglossaryentryposthook\@newglossaryentryposthook
9357 \renewcommand*{\@newglossaryentryposthook}{%
9358   \@gls@oldnewglossaryentryposthook
```

Store the access information:

```
9359   \expandafter
9360     \protected@xdef\csname glo@\@glo@label @access\endcsname{%
9361       \@glo@access}%
9362   \expandafter
9363     \protected@xdef\csname glo@\@glo@label @textaccess\endcsname{%
9364       \@glo@textaccess}%
9365   \expandafter
9366     \protected@xdef\csname glo@\@glo@label @firstaccess\endcsname{%
9367       \@glo@firstaccess}%
9368   \expandafter
9369     \protected@xdef\csname glo@\@glo@label @pluralaccess\endcsname{%
9370       \@glo@pluralaccess}%
9371   \expandafter
9372     \protected@xdef\csname glo@\@glo@label @firstpluralaccess\endcsname{%
9373       \@glo@firstpluralaccess}%
9374   \expandafter
9375     \protected@xdef\csname glo@\@glo@label @symbolaccess\endcsname{%
9376       \@glo@symbolaccess}%
9377   \expandafter
9378     \protected@xdef\csname glo@\@glo@label @symbolpluralaccess\endcsname{%
9379       \@glo@symbolpluralaccess}%
9380   \expandafter
9381     \protected@xdef\csname glo@\@glo@label @descaccess\endcsname{%
9382       \@glo@descaccess}%
9383   \expandafter
9384     \protected@xdef\csname glo@\@glo@label @descpluralaccess\endcsname{%
9385       \@glo@descpluralaccess}%
9386   \expandafter
9387     \protected@xdef\csname glo@\@glo@label @shortaccess\endcsname{%
9388       \@glo@shortaccess}%
9389   \expandafter
9390     \protected@xdef\csname glo@\@glo@label @shortpluralaccess\endcsname{%
9391       \@glo@shortpluralaccess}%
9392   \expandafter
9393     \protected@xdef\csname glo@\@glo@label @longaccess\endcsname{%
9394       \@glo@longaccess}%
9395   \expandafter
9396     \protected@xdef\csname glo@\@glo@label @longpluralaccess\endcsname{%
9397       \@glo@longpluralaccess}%
9398 }
```

7.2 Accessing Replacement Text

`\glsentryaccess` Get the value of the access key for the entry with the given label:

```
9399 \newcommand*{\glsentryaccess}[1]{%
9400   \@gls@entry@field{#1}{access}%
9401 }
```

`\glsentrytextaccess` Get the value of the textaccess key for the entry with the given label:

```
9402 \newcommand*{\glsentrytextaccess}[1]{%
9403   \@gls@entry@field{#1}{textaccess}%
9404 }
```

`\glsentryfirstaccess` Get the value of the firstaccess key for the entry with the given label:

```
9405 \newcommand*{\glsentryfirstaccess}[1]{%
9406   \@gls@entry@field{#1}{firstaccess}%
9407 }
```

`\glsentrypluralaccess` Get the value of the pluralaccess key for the entry with the given label:

```
9408 \newcommand*{\glsentrypluralaccess}[1]{%
9409   \@gls@entry@field{#1}{pluralaccess}%
9410 }
```

`\glsentryfirstpluralaccess` Get the value of the firstpluralaccess key for the entry with the given label:

```
9411 \newcommand*{\glsentryfirstpluralaccess}[1]{%
9412   \csname glo@#1@firstpluralaccess\endcsname
9413 }
```

`\glsentrysymbolaccess` Get the value of the symbolaccess key for the entry with the given label:

```
9414 \newcommand*{\glsentrysymbolaccess}[1]{%
9415   \@gls@entry@field{#1}{symbolaccess}%
9416 }
```

`\glsentrysymbolpluralaccess` Get the value of the symbolpluralaccess key for the entry with the given label:

```
9417 \newcommand*{\glsentrysymbolpluralaccess}[1]{%
9418   \@gls@entry@field{#1}{symbolpluralaccess}%
9419 }
```

`\glsentrydescaccess` Get the value of the descriptionaccess key for the entry with the given label:

```
9420 \newcommand*{\glsentrydescaccess}[1]{%
9421   \@gls@entry@field{#1}{descaccess}%
9422 }
```

`\glsentrydescpluralaccess` Get the value of the descriptionpluralaccess key for the entry with the given label:

```
9423 \newcommand*{\glsentrydescpluralaccess}[1]{%
9424   \@gls@entry@field{#1}{descaccess}%
9425 }
```


`\glsentryshortaccess` Get the value of the shortaccess key for the entry with the given label:

```
9426 \newcommand*{\glsentryshortaccess}[1]{%
9427   \@gls@entry@field{#1}{shortaccess}%
9428 }
```

`\glsentryshortpluralaccess` Get the value of the shortpluralaccess key for the entry with the given label:

```
9429 \newcommand*{\glsentryshortpluralaccess}[1]{%
9430   \@gls@entry@field{#1}{shortpluralaccess}%
9431 }
```

`\glsentrylongaccess` Get the value of the longaccess key for the entry with the given label:

```
9432 \newcommand*{\glsentrylongaccess}[1]{%
9433   \@gls@entry@field{#1}{longaccess}%
9434 }
```

`\glsentrylongpluralaccess` Get the value of the longpluralaccess key for the entry with the given label:

```
9435 \newcommand*{\glsentrylongpluralaccess}[1]{%
9436   \@gls@entry@field{#1}{longpluralaccess}%
9437 }
```

`\glsaccsupp` `\glsaccsupp{<replacement text>}{<text>}`

This can be redefined to use E or Alt instead of ActualText. (I don't have the software to test the E or Alt options.)

```
9438 \newcommand*{\glsaccsupp}[2]{%
9439   \BeginAccSupp{ActualText=#1}#2\EndAccSupp{}%
9440 }
```

`\xglsaccsupp` Fully expands replacement text before calling `\glsaccsupp`

```
9441 \newcommand*{\xglsaccsupp}[2]{%
9442   \protected@edef\@gls@replacementtext{#1}%
9443   \expandafter\glsaccsupp\expandafter{\@gls@replacementtext}{#2}%
9444 }
```

`@gls@access@display`

```
9445 \newcommand*{\@gls@access@display}[2]{%
9446   \protected@edef\@glo@access{#2}%
9447   \ifx\@glo@access\@gls@noaccess
9448     #1%
9449   \else
9450     \xglsaccsupp{\@glo@access}{#1}%
9451   \fi
9452 }
```

`\glsnameaccessdisplay` Displays the first argument with the accessibility text for the entry with the label given by the second argument (if set).

```
9453 \DeclareRobustCommand*{\glsnameaccessdisplay}[2]{%
9454   \@gls@access@display{#1}{\glsentryaccess{#2}}%
9455 }
```

lstextaccessdisplay As above but for the textaccess replacement text.

```

9456 \DeclareRobustCommand*\glstextaccessdisplay}[2]{%
9457   \@gls@access@display{#1}{\glsentrytextaccess{#2}}%
9458 }

```

pluralaccessdisplay As above but for the pluralaccess replacement text.

```

9459 \DeclareRobustCommand*\glspluralaccessdisplay}[2]{%
9460   \@gls@access@display{#1}{\glsentrypluralaccess{#2}}%
9461 }

```

sfirstaccessdisplay As above but for the firstaccess replacement text.

```

9462 \DeclareRobustCommand*\glsfirstaccessdisplay}[2]{%
9463   \@gls@access@display{#1}{\glsentryfirstaccess{#2}}%
9464 }

```

pluralaccessdisplay As above but for the firstpluralaccess replacement text.

```

9465 \DeclareRobustCommand*\glsfirstpluralaccessdisplay}[2]{%
9466   \@gls@access@display{#1}{\glsentryfirstpluralaccess{#2}}%
9467 }

```

symbolaccessdisplay As above but for the symbolaccess replacement text.

```

9468 \DeclareRobustCommand*\glssymbolaccessdisplay}[2]{%
9469   \@gls@access@display{#1}{\glsentrysymbolaccess{#2}}%
9470 }

```

pluralaccessdisplay As above but for the symbolpluralaccess replacement text.

```

9471 \DeclareRobustCommand*\glssymbolpluralaccessdisplay}[2]{%
9472   \@gls@access@display{#1}{\glsentrysymbolpluralaccess{#2}}%
9473 }

```

ptionaccessdisplay As above but for the descriptionaccess replacement text.

```

9474 \DeclareRobustCommand*\glsdescriptionaccessdisplay}[2]{%
9475   \@gls@access@display{#1}{\glsentrydescaccess{#2}}%
9476 }

```

pluralaccessdisplay As above but for the descriptionpluralaccess replacement text.

```

9477 \DeclareRobustCommand*\glsdescriptionpluralaccessdisplay}[2]{%
9478   \@gls@access@display{#1}{\glsentrydescpluralaccess{#2}}%
9479 }

```

sshortaccessdisplay As above but for the shortaccess replacement text.

```

9480 \DeclareRobustCommand*\glsshortaccessdisplay}[2]{%
9481   \@gls@access@display{#1}{\glsentryshortaccess{#2}}%
9482 }

```

pluralaccessdisplay As above but for the shortpluralaccess replacement text.

```

9483 \DeclareRobustCommand*\glsshortpluralaccessdisplay}[2]{%
9484   \@gls@access@display{#1}{\glsentryshortpluralaccess{#2}}%
9485 }

```

`longaccessdisplay` As above but for the `longaccess` replacement text.

```

9486 \DeclareRobustCommand*\glslongaccessdisplay}[2]{%
9487   \@gls@access@display{#1}{\glsentrylongaccess{#2}}%
9488 }

```

`pluralaccessdisplay` As above but for the `longpluralaccess` replacement text.

```

9489 \DeclareRobustCommand*\glslongpluralaccessdisplay}[2]{%
9490   \@gls@access@display{#1}{\glsentrylongpluralaccess{#2}}%
9491 }

```

`\glsaccessdisplay` Gets the replacement text corresponding to the named key given by the first argument and calls the appropriate command defined above.

```

9492 \DeclareRobustCommand*\glsaccessdisplay}[3]{%
9493   \@ifundefined{gls#1accessdisplay}%
9494   {%
9495     \PackageError{glossaries-accsupp}{No accessibility support
9496       for key ‘#1’}{%
9497     }%
9498   {%
9499     \csname gls#1accessdisplay\endcsname{#2}{#3}%
9500   }%
9501 }

```

`ls@default@entryfmt` Redefine the default entry format to use accessibility information

```

9502 \renewcommand*\@@gls@default@entryfmt}[2]{%
9503   \ifdefempty\glscustomtext
9504   {%
9505     \glsifplural
9506     {%

```

Plural form

```

9507       \glscapscase
9508       {%

```

Don't adjust case

```

9509       \ifglsused\glslabel
9510       {%

```

Subsequent use

```

9511         #2{\glspluralaccessdisplay
9512           {\glsentryplural{\glslabel}}{\glslabel}}%
9513         {\glsdescriptionpluralaccessdisplay
9514           {\glsentrydescplural{\glslabel}}{\glslabel}}%
9515         {\glssymbolpluralaccessdisplay
9516           {\glsentrysymbolplural{\glslabel}}{\glslabel}}
9517         {\glsinsert}%
9518       }%
9519     }%

```

First use

```

9520      #1{\glsfirstpluralaccessdisplay
9521          {\glsentryfirstplural{\glslabel}}{\glslabel}}%
9522          {\glsdescriptionpluralaccessdisplay
9523              {\glsentrydescplural{\glslabel}}{\glslabel}}%
9524          {\glssymbolpluralaccessdisplay
9525              {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
9526          {\glsinsert}}%
9527      }%
9528  }%
9529  {%

```

Make first letter upper case

```

9530      \ifglsused\glslabel
9531      {%

```

Subsequent use.

```

9532      #2{\glspluralaccessdisplay
9533          {\Glsentryplural{\glslabel}}{\glslabel}}%
9534          {\glsdescriptionpluralaccessdisplay
9535              {\glsentrydescplural{\glslabel}}{\glslabel}}%
9536          {\glssymbolpluralaccessdisplay
9537              {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
9538          {\glsinsert}}%
9539      }%
9540  {%

```

First use

```

9541      #1{\glsfirstpluralaccessdisplay
9542          {\Glsentryfirstplural{\glslabel}}{\glslabel}}%
9543          {\glsdescriptionpluralaccessdisplay
9544              {\glsentrydescplural{\glslabel}}{\glslabel}}%
9545          {\glssymbolpluralaccessdisplay
9546              {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
9547          {\glsinsert}}%
9548      }%
9549  }%
9550  {%

```

Make all upper case

```

9551      \ifglsused\glslabel
9552      {%

```

Subsequent use

```

9553      \MakeUppercase{%
9554      #2{\glspluralaccessdisplay
9555          {\glsentryplural{\glslabel}}{\glslabel}}%
9556          {\glsdescriptionpluralaccessdisplay
9557              {\glsentrydescplural{\glslabel}}{\glslabel}}%
9558          {\glssymbolpluralaccessdisplay
9559              {\glsentrysymbolplural{\glslabel}}{\glslabel}}%

```

```

9560          {\glsinsert}}}%
9561      }%
9562      {%

```

First use

```

9563      \MakeUppercase{%
9564          #1{\glsfirstpluralaccessdisplay
9565              {\glsentryfirstplural{\glslabel}}{\glslabel}}}%
9566              {\glsdescriptionpluralaccessdisplay
9567                  {\glsentrydescplural{\glslabel}}{\glslabel}}}%
9568              {\glsymbolpluralaccessdisplay
9569                  {\glsentrysymbolplural{\glslabel}}{\glslabel}}}%
9570          {\glsinsert}}}%
9571      }%
9572      }%
9573      }%
9574      {%

```

Singular form

```

9575      \glscapscase
9576      {%

```

Don't adjust case

```

9577      \ifglsused\glslabel
9578      {%

```

Subsequent use

```

9579      #2{\glstextaccessdisplay
9580          {\glsentrytext{\glslabel}}{\glslabel}}}%
9581          {\glsdescriptionaccessdisplay
9582              {\glsentrydesc{\glslabel}}{\glslabel}}}%
9583              {\glsymbolaccessdisplay
9584                  {\glsentrysymbol{\glslabel}}{\glslabel}}}%
9585          {\glsinsert}}}%
9586      }%
9587      {%

```

First use

```

9588      #1{\glsfirstaccessdisplay
9589          {\glsentryfirst{\glslabel}}{\glslabel}}}%
9590          {\glsdescriptionaccessdisplay
9591              {\glsentrydesc{\glslabel}}{\glslabel}}}%
9592              {\glsymbolaccessdisplay
9593                  {\glsentrysymbol{\glslabel}}{\glslabel}}}%
9594          {\glsinsert}}}%
9595      }%
9596      }%
9597      {%

```

Make first letter upper case

```

9598      \ifglsused\glslabel
9599      {%

```

Subsequent use

```

9600      #2{\glstextaccessdisplay
9601          {\Glentrytext{\glslabel}}{\glslabel}}%
9602          {\glsdescriptionaccessdisplay
9603              {\glentrydesc{\glslabel}}{\glslabel}}%
9604          {\glssymbolaccessdisplay
9605              {\glentrysymbol{\glslabel}}{\glslabel}}%
9606          {\glinsert}}%
9607      }%
9608      {%

```

First use

```

9609      #1{\glfirstaccessdisplay
9610          {\Glentryfirst{\glslabel}}{\glslabel}}%
9611          {\glsdescriptionaccessdisplay
9612              {\glentrydesc{\glslabel}}{\glslabel}}%
9613          {\glssymbolaccessdisplay
9614              {\glentrysymbol{\glslabel}}{\glslabel}}%
9615          {\glinsert}}%
9616      }%
9617      }%
9618      {%

```

Make all upper case

```

9619      \ifglused\glslabel
9620      {%

```

Subsequent use

```

9621      \MakeUppercase{%
9622      #2{\glstextaccessdisplay
9623          {\glentrytext{\glslabel}}{\glslabel}}%
9624          {\glsdescriptionaccessdisplay
9625              {\glentrydesc{\glslabel}}{\glslabel}}%
9626          {\glssymbolaccessdisplay
9627              {\glentrysymbol{\glslabel}}{\glslabel}}%
9628          {\glinsert}}%
9629      }%
9630      {%

```

First use

```

9631      \MakeUppercase{%
9632      #1{\glfirstaccessdisplay
9633          {\glentryfirst{\glslabel}}{\glslabel}}%
9634          {\glsdescriptionaccessdisplay
9635              {\glentrydesc{\glslabel}}{\glslabel}}%
9636          {\glssymbolaccessdisplay
9637              {\glentrysymbol{\glslabel}}{\glslabel}}%
9638          {\glinsert}}%
9639      }%
9640      }%
9641      }%

```

```

9642 }%
9643 {%
    Custom text provided in \glsdisp
9644 \ifglsused{\glslabel}%
9645 {%
    Subsequent use
9646 #2{\glscustomtext}%
9647 {\glsdescriptionaccessdisplay
9648   {\glsentrydesc{\glslabel}}{\glslabel}}%
9649 {\glssymbolaccessdisplay
9650   {\glsentrysymbol{\glslabel}}{\glslabel}}%
9651 {\glsinsert}%
9652 }%
9653 {%
    First use
9654 #1{\glscustomtext}%
9655 {\glsdescriptionaccessdisplay
9656   {\glsentrydesc{\glslabel}}{\glslabel}}%
9657 {\glssymbolaccessdisplay
9658   {\glsentrysymbol{\glslabel}}{\glslabel}}%
9659 {\glsinsert}%
9660 }%
9661 }%
9662 }

```

`\glsgenentryfmt` Redefine to use accessibility information.

```

9663 \renewcommand*{\glsgenentryfmt}{%
9664   \ifdefempty\glscustomtext
9665   {%
9666     \glsifplural
9667     {%
        Plural form
9668     \glscapscase
9669     {%
        Don't adjust case
9670     \ifglsused\glslabel
9671     {%
        Subsequent use
9672     \glspluralaccessdisplay
9673     {\glsentryplural{\glslabel}}{\glslabel}%
9674     \glsinsert
9675     }%
9676     {%
        First use
9677     \glsfirstpluralaccessdisplay

```

```

9678             {\glsentryfirstplural{\glslabel}}{\glslabel}%
9679         \glsinsert
9680     }%
9681 }%
9682 {%

```

Make first letter upper case

```

9683     \ifglsused\glslabel
9684     {%

```

Subsequent use.

```

9685     \glspluralaccessdisplay
9686     {\Glsentryplural{\glslabel}}{\glslabel}%
9687     \glsinsert
9688 }%
9689 {%

```

First use

```

9690     \glsfirstpluralaccessdisplay
9691     {\Glsentryfirstplural{\glslabel}}{\glslabel}%
9692     \glsinsert
9693 }%
9694 }%
9695 {%

```

Make all upper case

```

9696     \ifglsused\glslabel
9697     {%

```

Subsequent use

```

9698     \glspluralaccessdisplay
9699     {\mfirstucMakeUppercase{\glsentryplural{\glslabel}}}%
9700     {\glslabel}%
9701     \mfirstucMakeUppercase{\glsinsert}%
9702 }%
9703 {%

```

First use

```

9704     \glsfirstpluralaccessdisplay
9705     {\mfirstucMakeUppercase{\glsentryfirstplural{\glslabel}}}%
9706     {\glslabel}%
9707     \mfirstucMakeUppercase{\glsinsert}%
9708 }%
9709 }%
9710 }%
9711 {%

```

Singular form

```

9712     \glscapscase
9713     {%

```

Don't adjust case


```

9714      \ifglsused\glslabel
9715      {%

```

Subsequent use

```

9716      \glstextaccessdisplay{\glsentrytext{\glslabel}}{\glslabel}%
9717      \glsinsert
9718      }%
9719      {%

```

First use

```

9720      \glsfirstaccessdisplay{\glsentryfirst{\glslabel}}{\glslabel}%
9721      \glsinsert
9722      }%
9723      }%
9724      {%

```

Make first letter upper case

```

9725      \ifglsused\glslabel
9726      {%

```

Subsequent use

```

9727      \glstextaccessdisplay{\Glsentrytext{\glslabel}}{\glslabel}%
9728      \glsinsert
9729      }%
9730      {%

```

First use

```

9731      \glsfirstaccessdisplay{\Glsentryfirst{\glslabel}}{\glslabel}%
9732      \glsinsert
9733      }%
9734      }%
9735      {%

```

Make all upper case

```

9736      \ifglsused\glslabel
9737      {%

```

Subsequent use

```

9738      \glstextaccessdisplay
9739      {\mfirstucMakeUppercase{\glsentrytext{\glslabel}}}{\glslabel}%
9740      \mfirstucMakeUppercase{\glsinsert}%
9741      }%
9742      {%

```

First use

```

9743      \glsfirstaccessdisplay
9744      {\mfirstucMakeUppercase{\glsentryfirst{\glslabel}}}{\glslabel}%
9745      \mfirstucMakeUppercase{\glsinsert}%
9746      }%
9747      }%
9748      }%
9749      }%
9750      {%

```

Custom text provided in `\glsdisp`. (The insert should be empty at this point.)
The accessibility information, if required, will have to be explicitly included in
the custom text.

```
9751 \glscustomtext\glsinsert
9752 }%
9753 }
```

`\glsngenacfmt` Redefine to include accessibility information.

```
9754 \renewcommand*{\glsngenacfmt}{%
9755 \ifdefempty\glscustomtext
9756 {%
9757 \ifglused\glslabel
9758 {%
```

Subsequent use:

```
9759 \glsifplural
9760 {%
```

Subsequent plural form:

```
9761 \glscapscase
9762 {%
```

Subsequent plural form, don't adjust case:

```
9763 \acronymfont
9764 {\glsshortpluralaccessdisplay
9765 {\glentryshortpl{\glslabel}}{\glslabel}}%
9766 \glsinsert
9767 }%
9768 {%
```

Subsequent plural form, make first letter upper case:

```
9769 \acronymfont
9770 {\glsshortpluralaccessdisplay
9771 {\Glsentryshortpl{\glslabel}}{\glslabel}}%
9772 \glsinsert
9773 }%
9774 {%
```

Subsequent plural form, all caps:

```
9775 \mfirstucMakeUppercase
9776 {\acronymfont
9777 {\glsshortpluralaccessdisplay
9778 {\glentryshortpl{\glslabel}}{\glslabel}}%
9779 \glsinsert}%
9780 }%
9781 }%
9782 {%
```

Subsequent singular form

```
9783 \glscapscase
9784 {%
```

Subsequent singular form, don't adjust case:

```
9785      \acronymfont
9786      {\glsshortaccessdisplay{\glsentryshort{\glslabel}}{\glslabel}}%
9787      \glsinsert
9788      }%
9789      {%
```

Subsequent singular form, make first letter upper case:

```
9790      \acronymfont
9791      {\glsshortaccessdisplay{\Glsentryshort{\glslabel}}{\glslabel}}%
9792      \glsinsert
9793      }%
9794      {%
```

Subsequent singular form, all caps:

```
9795      \mfirstucMakeUppercase
9796      {\acronymfont{%
9797      \glsshortaccessdisplay{\glsentryshort{\glslabel}}{\glslabel}}%
9798      \glsinsert}%
9799      }%
9800      }%
9801      }%
9802      {%
```

First use:

```
9803      \glsifplural
9804      {%
```

First use plural form:

```
9805      \glscapscase
9806      {%
```

First use plural form, don't adjust case:

```
9807      \genplacrfullformat{\glslabel}{\glsinsert}%
9808      }%
9809      {%
```

First use plural form, make first letter upper case:

```
9810      \Genplacrfullformat{\glslabel}{\glsinsert}%
9811      }%
9812      {%
```

First use plural form, all caps:

```
9813      \mfirstucMakeUppercase
9814      {\genplacrfullformat{\glslabel}{\glsinsert}}%
9815      }%
9816      }%
9817      {%
```

First use singular form

```
9818      \glscapscase
9819      {%
```

First use singular form, don't adjust case:

```
9820      \genacrfullformat{\glslabel}{\glsinsert}%
9821      }%
9822      {%
```

First use singular form, make first letter upper case:

```
9823      \Genacrfullformat{\glslabel}{\glsinsert}%
9824      }%
9825      {%
```

First use singular form, all caps:

```
9826      \mfirstucMakeUppercase
9827      {\genacrfullformat{\glslabel}{\glsinsert}}%
9828      }%
9829      }%
9830      }%
9831      }%
9832      {%
```

User supplied text. (The insert should be empty at this point.) The accessibility information, if required, will have to be explicitly included in the custom text.

```
9833      \glscustomtext
9834      }%
9835 }
```

`\genacrfullformat` Redefine to include accessibility information.

```
9836 \renewcommand*{\genacrfullformat}[2]{%
9837   \glslongaccessdisplay{\glsentrylong{#1}}{#1}#2\space
9838   (\glsshortaccessdisplay{\protect\firstacronymfont{\glsentryshort{#1}}}{#1})%
9839 }
```

`\Genacrfullformat` Redefine to include accessibility information.

```
9840 \renewcommand*{\Genacrfullformat}[2]{%
9841   \glslongaccessdisplay{\Glsentrylong{#1}}{#1}#2\space
9842   (\glsshortaccessdisplay{\protect\firstacronymfont{\Glsentryshort{#1}}}{#1})%
9843 }
```

`\genplacrfullformat` Redefine to include accessibility information.

```
9844 \renewcommand*{\genplacrfullformat}[2]{%
9845   \glslongpluralaccessdisplay{\glsentrylongpl{#1}}{#1}#2\space
9846   (\glsshortpluralaccessdisplay
9847     {\protect\firstacronymfont{\glsentryshortpl{#1}}}{#1})%
9848 }
```

`\Genplacrfullformat` Redefine to include accessibility information.

```
9849 \renewcommand*{\Genplacrfullformat}[2]{%
9850   \glslongpluralaccessdisplay{\Glsentrylongpl{#1}}{#1}#2\space
9851   (\glsshortpluralaccessdisplay
9852     {\protect\firstacronymfont{\Glsentryshortpl{#1}}}{#1})%
9853 }
```

\@acrshort

```

9854 \def\@acrshort#1#2[#3]{%
9855   \glsdoifexists{#2}%
9856   {%
9857     \let\do@gl@link@checkfirsthyper\relax

9858     \let\glsifplural\@secondoftwo
9859     \let\glscapscase\@firstofthree
9860     \let\glsinsert\@empty
9861     \def\glscustomtext{%
9862       \acronymfont{\glsshortaccessdisplay{\glsentryshort{#2}}{#2}}#3%
9863     }%

    Call \@gl@link
9864     \@gl@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
9865   }%
9866 }
```

\@Acrshort

```

9867 \def\@Acrshort#1#2[#3]{%
9868   \glsdoifexists{#2}%
9869   {%
9870     \let\do@gl@link@checkfirsthyper\relax

9871     \let\glsifplural\@secondoftwo
9872     \let\glscapscase\@secondofthree
9873     \let\glsinsert\@empty
9874     \def\glscustomtext{%
9875       \acronymfont{\glsshortaccessdisplay{\Glsentryshort{#2}}{#2}}#3%
9876     }%

    Call \@gl@link
9877     \@gl@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
9878   }%
9879 }
```

\@ACRshort

```

9880 \def\@ACRshort#1#2[#3]{%
9881   \glsdoifexists{#2}%
9882   {%
9883     \let\do@gl@link@checkfirsthyper\relax

9884     \let\glsifplural\@secondoftwo
9885     \let\glscapscase\@thirdofthree
9886     \let\glsinsert\@empty
9887     \def\glscustomtext{%
9888       \acronymfont{\glsshortaccessdisplay
9889         {\MakeUppercase{\glsentryshort{#2}}}{#2}}#3%
9890     }%
9891 }
```

```

Call \@gls@link
9891   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
9892   }%
9893 }

```

\@acrlong

```

9894 \def\@acrlong#1#2[#3]{%
9895   \glsdoifexists{#2}%
9896   {%
9897     \let\do@gls@link@checkfirsthyper\relax

9898     \let\glsifplural\@secondoftwo
9899     \let\glscapscase\@firstofthree
9900     \let\glsinsert\@empty
9901     \def\glscustomtext{%
9902       \acronymfont{\glslongaccessdisplay{\glsentrylong{#2}}{#2}}#3%
9903     }%

```

```

Call \@gls@link
9904   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
9905   }%
9906 }

```

\@Acrlong

```

9907 \def\@Acrlong#1#2[#3]{%
9908   \glsdoifexists{#2}%
9909   {%
9910     \let\do@gls@link@checkfirsthyper\relax

9911     \let\glsifplural\@secondoftwo
9912     \let\glscapscase\@firstofthree
9913     \let\glsinsert\@empty
9914     \def\glscustomtext{%
9915       \acronymfont{\glslongaccessdisplay{\Glsentrylong{#2}}{#2}}#3%
9916     }%

```

```

Call \@gls@link
9917   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
9918   }%
9919 }

```

\@ACRlong

```

9920 \def\@ACRlong#1#2[#3]{%
9921   \glsdoifexists{#2}%
9922   {%
9923     \let\do@gls@link@checkfirsthyper\relax

9924     \let\glsifplural\@secondoftwo
9925     \let\glscapscase\@firstofthree
9926     \let\glsinsert\@empty

```

```

9927 \def\glscustomtext{%
9928 \acronymfont{\glslongaccessdisplay{%
9929 \MakeUppercase{\glsentrylong{#2}}}{#2}#3}%
9930 }%

Call \@gls@link
9931 \@gls@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
9932 }%
9933 }

```

7.3 Displaying the Glossary

We need to redefine the way the glossary entries are formatted to include the accessibility support. The predefined glossary styles use `\glossentryname`, `\glossentrydesc` and `\glossentrysymbol`, but we need to provide compatibility with earlier versions in case users have defined their own styles using `\accsuppglossaryentryfield` and `\accsuppglossarysubentryfield`.

Now redefine `\glossentryname`, `\glossentrydesc` and `\glossentrysymbol` etc so they use the accessibility stuff.

```

9934 \renewcommand*{\glossentryname}[1]{%
9935 \glsdoifexists{#1}%
9936 {%
9937 \glsnamefont{\glsnameaccessdisplay{\glsentryname{#1}}{#1}}%
9938 }%
9939 }

9940 \renewcommand*{\glossentrydesc}[1]{%
9941 \glsdoifexists{#1}%
9942 {%
9943 \glsnamefont{\glsnameaccessdisplay{\Glsentryname{#1}}{#1}}%
9944 }%
9945 }

9946 \renewcommand*{\glossentrydesc}[1]{%
9947 \glsdoifexists{#1}%
9948 {%
9949 \glsdescriptionaccessdisplay{\glsentrydesc{#1}}{#1}%
9950 }%
9951 }

9952 \renewcommand*{\Glossentrydesc}[1]{%
9953 \glsdoifexists{#1}%
9954 {%
9955 \glsdescriptionaccessdisplay{\Glsentrydesc{#1}}{#1}%
9956 }%
9957 }

9958 \renewcommand*{\glossentrysymbol}[1]{%
9959 \glsdoifexists{#1}%
9960 {%

```

```

9961 \glssymbolaccessdisplay{\glentrysymbol{#1}}{#1}%
9962 }%
9963 }

9964 \renewcommand*{\Glossentrysymbol}[1]{%
9965 \glsoifexists{#1}%
9966 {%
9967 \glssymbolaccessdisplay{\Glentrysymbol{#1}}{#1}%
9968 }%
9969 }

```

pglossaryentryfield

```

9970 \newcommand*{\accsuppglossaryentryfield}[5]{%
9971 \glossaryentryfield{#1}%
9972 {\glnameaccessdisplay{#2}{#1}}%
9973 {\gldescriptionaccessdisplay{#3}{#1}}%
9974 {\glssymbolaccessdisplay{#4}{#1}}{#5}%
9975 }

```

glossarysubentryfield

```

9976 \newcommand*{\accsuppglossarysubentryfield}[6]{%
9977 \glossarysubentryfield{#1}{#2}%
9978 {\glnameaccessdisplay{#3}{#2}}%
9979 {\gldescriptionaccessdisplay{#4}{#2}}%
9980 {\glssymbolaccessdisplay{#5}{#2}}{#6}%
9981 }

```

7.4 Acronyms

Redefine acronym styles provided by glossaries:

long-short *<long>* (*<short>*) acronym style.

```

9982 \renewacronymstyle{long-short}%
9983 {%

```

Check for long form in case this is a mixed glossary.

```

9984 \ifglshaslong{\glslabel}{\glsgenacfmt}{\glsgenentryfmt}%
9985 }%
9986 {%
9987 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
9988 \renewcommand*{\genacrfullformat}[2]{%
9989 \glslongaccessdisplay{\glentrylong{##1}}{##1}##2\space
9990 (\glsshortaccessdisplay
9991 {\protect\firstacronymfont{\glentryshort{##1}}}{##1})%
9992 }%
9993 \renewcommand*{\Genacrfullformat}[2]{%
9994 \glslongaccessdisplay{\Glentrylong{##1}}{##1}##2\space
9995 (\glsshortaccessdisplay
9996 {\protect\firstacronymfont{\glentryshort{##1}}}{##1})%
9997 }%

```



```

9998 \renewcommand*{\genplacrfullformat}[2]{%
9999 \glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1}##2\space
10000 (\glsshortpluralaccessdisplay
10001 {\protect\firstacronymfont{\glsentryshortpl{##1}}}{##1})%
10002 }%
10003 \renewcommand*{\Genplacrfullformat}[2]{%
10004 \glslongpluralaccessdisplay{\Glsentrylongpl{##1}}{##1}##2\space
10005 (\glsshortpluralaccessdisplay
10006 {\protect\firstacronymfont{\Glsentryshortpl{##1}}}{##1})%
10007 }%
10008 \renewcommand*{\acronymentry}[1]{%
10009 \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}}
10010 \renewcommand*{\acronymsort}[2]{##1}%
10011 \renewcommand*{\acronymfont}[1]{##1}%
10012 \renewcommand*{\firstacronymfont}[1]{\acronymfont{##1}}%
10013 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
10014 }

```

short-long <short> (<long>) acronym style.

```

10015 \renewacronymstyle{short-long}%
10016 {%
10017 \ifglshaslong{\glslabel}{\glsngenacfmt}{\glsngenentryfmt}%
10018 }%
10019 {%
10020 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
10021 \renewcommand*{\genacrfullformat}[2]{%
10022 \glsshortaccessdisplay
10023 {\protect\firstacronymfont{\glsentryshort{##1}}}{##1}##2\space
10024 (\glslongaccessdisplay{\glsentrylong{##1}}{##1})%
10025 }%
10026 \renewcommand*{\Genacrfullformat}[2]{%
10027 \glsshortaccessdisplay
10028 {\protect\firstacronymfont{\Glsentryshort{##1}}}{##1}##2\space
10029 (\glslongaccessdisplay{\Glsentrylong{##1}}{##1})%
10030 }%
10031 \renewcommand*{\genplacrfullformat}[2]{%
10032 \glsshortpluralaccessdisplay
10033 {\protect\firstacronymfont{\glsentryshortpl{##1}}}{##1}##2\space
10034 (\glslongpluralaccessdisplay
10035 {\glsentrylongpl{##1}}{##1})%
10036 }%
10037 \renewcommand*{\Genplacrfullformat}[2]{%
10038 \glsshortpluralaccessdisplay
10039 {\protect\firstacronymfont{\Glsentryshortpl{##1}}}{##1}##2\space
10040 (\glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1})%
10041 }%
10042 \renewcommand*{\acronymentry}[1]{%
10043 \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}}%

```

```

10044 \renewcommand*{\acronymsort}[2]{##1}%
10045 \renewcommand*{\acronymfont}[1]{##1}%
10046 \renewcommand*{\firstacronymfont}[1]{\acronymfont{##1}}%
10047 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
10048 }

```

long-short-desc *⟨long⟩* (*⟨short⟩*) acronym style that has an accompanying description (which the user needs to supply).

```

10049 \renewacronymstyle{long-short-desc}%
10050 {%
10051   \GlsUseAcrEntryDisplayStyle{long-short}%
10052 }%
10053 {%
10054   \GlsUseAcrStyleDefs{long-short}%
10055   \renewcommand*{\GenericAcronymFields}{}%
10056   \renewcommand*{\acronymsort}[2]{##2}%
10057   \renewcommand*{\acronymentry}[1]{%
10058     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10059     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10060 }

```

long-sc-short-desc *⟨long⟩* (\textsc{⟨short⟩}) acronym style that has an accompanying description (which the user needs to supply).

```

10061 \renewacronymstyle{long-sc-short-desc}%
10062 {%
10063   \GlsUseAcrEntryDisplayStyle{long-sc-short}%
10064 }%
10065 {%
10066   \GlsUseAcrStyleDefs{long-sc-short}%
10067   \renewcommand*{\GenericAcronymFields}{}%
10068   \renewcommand*{\acronymsort}[2]{##2}%
10069   \renewcommand*{\acronymentry}[1]{%
10070     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10071     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10072 }

```

long-sm-short-desc *⟨long⟩* (\textsmaller{⟨short⟩}) acronym style that has an accompanying description (which the user needs to supply).

```

10073 \renewacronymstyle{long-sm-short-desc}%
10074 {%
10075   \GlsUseAcrEntryDisplayStyle{long-sm-short}%
10076 }%
10077 {%
10078   \GlsUseAcrStyleDefs{long-sm-short}%
10079   \renewcommand*{\GenericAcronymFields}{}%
10080   \renewcommand*{\acronymsort}[2]{##2}%
10081   \renewcommand*{\acronymentry}[1]{%
10082     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10083     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%

```

10084 }

short-long-desc *<short>* (*{<long>}*) acronym style that has an accompanying description (which the user needs to supply).

```
10085 \renewacronymstyle{short-long-desc}%
10086 {%
10087   \GlsUseAcrEntryDispStyle{short-long}%
10088 }%
10089 {%
10090   \GlsUseAcrStyleDefs{short-long}%
10091   \renewcommand*{\GenericAcronymFields}{}%
10092   \renewcommand*{\acronymsort}[2]{##2}%
10093   \renewcommand*{\acronymentry}[1]{%
10094     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10095     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10096 }
```

sc-short-long-desc *<long>* (*\textsc{<short>}*) acronym style that has an accompanying description (which the user needs to supply).

```
10097 \renewacronymstyle{sc-short-long-desc}%
10098 {%
10099   \GlsUseAcrEntryDispStyle{sc-short-long}%
10100 }%
10101 {%
10102   \GlsUseAcrStyleDefs{sc-short-long}%
10103   \renewcommand*{\GenericAcronymFields}{}%
10104   \renewcommand*{\acronymsort}[2]{##2}%
10105   \renewcommand*{\acronymentry}[1]{%
10106     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10107     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10108 }
```

sm-short-long-desc *<long>* (*\textsmaller{<short>}*) acronym style that has an accompanying description (which the user needs to supply).

```
10109 \renewacronymstyle{sm-short-long-desc}%
10110 {%
10111   \GlsUseAcrEntryDispStyle{sm-short-long}%
10112 }%
10113 {%
10114   \GlsUseAcrStyleDefs{sm-short-long}%
10115   \renewcommand*{\GenericAcronymFields}{}%
10116   \renewcommand*{\acronymsort}[2]{##2}%
10117   \renewcommand*{\acronymentry}[1]{%
10118     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10119     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10120 }
```

dua *<long>* only acronym style.

10121 \renewacronymstyle{dua}%

10122 {%

Check for long form in case this is a mixed glossary.

10123 \ifdefempty\glscustomtext

10124 {%

10125 \ifglshaslong{\glslabel}%

10126 {%

10127 \glsifplural

10128 {%

Plural form:

10129 \glscapscase

10130 {%

Plural form, don't adjust case:

10131 \glslongpluralaccessdisplay{\glstentrylongpl{\glslabel}}{\glslabel}%

10132 \glsinsert

10133 }%

10134 {%

Plural form, make first letter upper case:

10135 \glslongpluralaccessdisplay{\Glstentrylongpl{\glslabel}}{\glslabel}%

10136 \glsinsert

10137 }%

10138 {%

Plural form, all caps:

10139 \glslongpluralaccessdisplay

10140 {\mfirstucMakeUppercase{\glstentrylongpl{\glslabel}}}{\glslabel}%

10141 \mfirstucMakeUppercase{\glsinsert}%

10142 }%

10143 }%

10144 {%

Singular form

10145 \glscapscase

10146 {%

Singular form, don't adjust case:

10147 \glslongaccessdisplay{\glstentrylong{\glslabel}}{\glslabel}\glsinsert

10148 }%

10149 {%

Subsequent singular form, make first letter upper case:

10150 \glslongaccessdisplay{\Glstentrylong{\glslabel}}{\glslabel}\glsinsert

10151 }%

10152 {%

Subsequent singular form, all caps:

10153 \glslongaccessdisplay

10154 {\mfirstucMakeUppercase

10155 {\glstentrylong{\glslabel}\glsinsert}}{\glslabel}%

```

10156         \mfirstucMakeUppercase{\glsinsert}%
10157     }%
10158 }%
10159 }%
10160 {%

```

Not an acronym:

```

10161     \glsгенentryfmt
10162 }%
10163 }%
10164 {\glscustomtext\glsinsert}%
10165 }%
10166 {%
10167 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
10168 \renewcommand*{\acrfullfmt}[3]{%
10169     \glslink[##1]{##2}{%
10170         \glslongaccessdisplay{\glsentrylong{##2}}{##2}##3\space
10171         (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##2}}}{##2}}}%
10172 \renewcommand*{\Acrfullfmt}[3]{%
10173     \glslink[##1]{##2}{%
10174         \glslongaccessdisplay{\Glsentrylong{##2}}{##2}##3\space
10175         (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##2}}}{##2}}}%
10176 \renewcommand*{\ACRfullfmt}[3]{%
10177     \glslink[##1]{##2}{%
10178         \glslongaccessdisplay
10179         {\mfirstucMakeUppercase{\glsentrylong{##2}}{##2}##3\space
10180         (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##2}}}{##2}}}}}%
10181 \renewcommand*{\acrfullplfmt}[3]{%
10182     \glslink[##1]{##2}{%
10183         \glslongpluralaccessdisplay
10184         {\glsentrylongpl{##2}}{##2}##3\space
10185         (\glsshortpluralaccessdisplay
10186         {\acronymfont{\glsentryshortpl{##2}}}{##2}}}%
10187 \renewcommand*{\Acrfullplfmt}[3]{%
10188     \glslink[##1]{##2}{%
10189         \glslongpluralaccessdisplay
10190         {\Glsentrylongpl{##2}}{##2}##3\space
10191         (\glsshortpluralaccessdisplay
10192         {\acronymfont{\glsentryshortpl{##2}}}{##2}}}%
10193 \renewcommand*{\ACRfullplfmt}[3]{%
10194     \glslink[##1]{##2}{%
10195         \glslongpluralaccessdisplay
10196         {\mfirstucMakeUppercase{\glsentrylongpl{##2}}{##2}##3\space
10197         (\glsshortpluralaccessdisplay
10198         {\acronymfont{\glsentryshortpl{##2}}}{##2}}}}}%
10199 \renewcommand*{\glsentryfull}[1]{%
10200     \glslongaccessdisplay{\glsentrylong{##1}}\space
10201     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})%
10202 }%
10203 \renewcommand*{\Glsentryfull}[1]{%

```

```

10204 \glslongaccessdisplay{\Glsentrylong{##1}}{##1}\space
10205 (\glsshortaccessdisplay{\acronymfont{\glentryshort{##1}}}{##1})%
10206 }%
10207 \renewcommand*{\glsentryfullpl}[1]{%
10208 \glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1}\space
10209 (\glsshortpluralaccessdisplay{\acronymfont{\glentryshortpl{##1}}}{##1})%
10210 }%
10211 \renewcommand*{\Glsentryfullpl}[1]{%
10212 \glslongpluralaccessdisplay{\Glsentrylongpl{##1}}{##1}\space
10213 (\glsshortpluralaccessdisplay{\acronymfont{\glentryshortpl{##1}}}{##1})%
10214 }%
10215 \renewcommand*{\acronymentry}[1]{%
10216 \glsshortaccessdisplay{\acronymfont{\glentryshort{##1}}}{##1}%
10217 \renewcommand*{\acronymsort}[2]{##1}%
10218 \renewcommand*{\acronymfont}[1]{##1}%
10219 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
10220 }

```

dua-desc *<long>* only acronym style with user-supplied description.

```

10221 \renewacronymstyle{dua-desc}%
10222 {%
10223 \GlsUseAcrEntryDispStyle{dua}%
10224 }%
10225 {%
10226 \GlsUseAcrStyleDefs{dua}%
10227 \renewcommand*{\GenericAcronymFields}{}%
10228 \renewcommand*{\acronymentry}[1]{%
10229 \glslongaccessdisplay{\acronymfont{\glentrylong{##1}}}{##1}%
10230 \renewcommand*{\acronymsort}[2]{##2}%
10231 }%

```

footnote *<short>*\footnote{*<long>*} acronym style.

```

10232 \renewacronymstyle{footnote}%
10233 {%

```

Check for long form in case this is a mixed glossary.

```

10234 \ifglshaslong{\glslabel}{\glsgenacfmt}{\glsgenentryfmt}%
10235 }%
10236 {%
10237 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%

```

Need to ensure hyperlinks are switched off on first use:

```

10238 \glshyperfirstfalse
10239 \renewcommand*{\genacrfullformat}[2]{%
10240 \glsshortaccessdisplay
10241 {\protect\firstacronymfont{\glentryshort{##1}}}{##1}##2%
10242 \protect\footnote{\glslongaccessdisplay{\glentrylong{##1}}{##1}}%
10243 }%
10244 \renewcommand*{\Genacrfullformat}[2]{%
10245 \glsshortaccessdisplay

```

```

10246     {\firstacronymfont{\Glsentryshort{##1}}}{##1}##2%
10247     \protect\footnote{\glslongaccessdisplay{\glssentrylong{##1}}{##1}}%
10248 }%
10249 \renewcommand*{\genplacrfullformat}[2]{%
10250     \glsshortpluralaccessdisplay
10251     {\protect\firstacronymfont{\glssentryshortpl{##1}}}{##1}##2%
10252     \protect\footnote{\glslongpluralaccessdisplay{\glssentrylongpl{##1}}{##1}}%
10253 }%
10254 \renewcommand*{\Genplacrfullformat}[2]{%
10255     \glsshortpluralaccessdisplay
10256     {\protect\firstacronymfont{\Glsentryshortpl{##1}}}{##1}##2%
10257     \protect\footnote{\glslongpluralaccessdisplay{\glssentrylongpl{##1}}{##1}}%
10258 }%
10259 \renewcommand*{\acronymentry}[1]{%
10260     \glsshortaccessdisplay{\acronymfont{\glssentryshort{##1}}}{##1}}%
10261 \renewcommand*{\acronymsort}[2]{##1}%
10262 \renewcommand*{\acronymfont}[1]{##1}%
10263 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%

```

Don't use footnotes for \acrfull:

```

10264 \renewcommand*{\acrfullfmt}[3]{%
10265     \glslink[##1]{##2}{%
10266         \glsshortaccessdisplay{\acronymfont{\glssentryshort{##2}}}{##2}##3\space
10267         (\glslongaccessdisplay{\glssentrylong{##2}}{##2})}%
10268 \renewcommand*{\Acrfullfmt}[3]{%
10269     \glslink[##1]{##2}{%
10270         \glsshortaccessdisplay{\acronymfont{\Glsentryshort{##2}}}{##2}##3\space
10271         (\glslongaccessdisplay{\glssentrylong{##2}}{##2})}%
10272 \renewcommand*{\ACRfullfmt}[3]{%
10273     \glslink[##1]{##2}{%
10274         \glsshortaccessdisplay
10275         {\mfirstucMakeUppercase
10276         {\acronymfont{\glssentryshort{##2}}}{##2}##3\space
10277         (\glslongaccessdisplay{\glssentrylong{##2}}{##2})}%
10278 \renewcommand*{\acrfullplfmt}[3]{%
10279     \glslink[##1]{##2}{%
10280         \glsshortpluralaccessdisplay
10281         {\acronymfont{\glssentryshortpl{##2}}}{##2}##3\space
10282         (\glslongpluralaccessdisplay{\glssentrylongpl{##2}}{##2})}%
10283 \renewcommand*{\Acrfullplfmt}[3]{%
10284     \glslink[##1]{##2}{%
10285         \glsshortpluralaccessdisplay
10286         {\acronymfont{\Glsentryshortpl{##2}}}{##2}##3\space
10287         (\glslongpluralaccessdisplay{\glssentrylongpl{##2}}{##2})}%
10288 \renewcommand*{\ACRfullplfmt}[3]{%
10289     \glslink[##1]{##2}{%
10290         \glsshortpluralaccessdisplay
10291         {\mfirstucMakeUppercase
10292         {\acronymfont{\glssentryshortpl{##2}}}{##2}##3\space
10293         (\glslongpluralaccessdisplay{\glssentrylongpl{##2}}{##2})}%

```

Similarly for \glsentryfull etc:

```

10294 \renewcommand*{\glsentryfull}[1]{%
10295   \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}\space
10296   (\glslongaccessdisplay{\glsentrylong{##1}}{##1})}%
10297 \renewcommand*{\Glsentryfull}[1]{%
10298   \glsshortaccessdisplay{\acronymfont{\Glsentryshort{##1}}}{##1}\space
10299   (\glslongaccessdisplay{\glsentrylong{##1}}{##1})}%
10300 \renewcommand*{\glsentryfullpl}[1]{%
10301   \glsshortpluralaccessdisplay
10302   {\acronymfont{\glsentryshortpl{##1}}}{##1}\space
10303   (\glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1})}%
10304 \renewcommand*{\Glsentryfullpl}[1]{%
10305   \glsshortpluralaccessdisplay
10306   {\acronymfont{\Glsentryshortpl{##1}}}{##1}\space
10307   (\glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1})}%
10308 }

```

footnote-sc \textsc{<short>}\footnote{<long>} acronym style.

```

10309 \renewacronymstyle{footnote-sc}%
10310 {%
10311   \GlsUseAcrEntryDispStyle{footnote}%
10312 }%
10313 {%
10314   \GlsUseAcrStyleDefs{footnote}%
10315   \renewcommand{\acronymentry}[1]{%
10316     \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}}
10317   \renewcommand{\acronymfont}[1]{\textsc{##1}}%
10318   \renewcommand*{\acrpluralsuffix}{\glstextup{\glspluralsuffix}}%
10319 }%

```

footnote-sm \textsmaller{<short>}\footnote{<long>} acronym style.

```

10320 \renewacronymstyle{footnote-sm}%
10321 {%
10322   \GlsUseAcrEntryDispStyle{footnote}%
10323 }%
10324 {%
10325   \GlsUseAcrStyleDefs{footnote}%
10326   \renewcommand{\acronymentry}[1]{%
10327     \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}}
10328   \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
10329   \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
10330 }%

```

footnote-desc <short>\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```

10331 \renewacronymstyle{footnote-desc}%
10332 {%
10333   \GlsUseAcrEntryDispStyle{footnote}%
10334 }%

```



```

10335 {%
10336   \GlsUseAcrStyleDefs{footnote}%
10337   \renewcommand*{\GenericAcronymFields}{}%
10338   \renewcommand*{\acronymsort}[2]{##2}%
10339   \renewcommand*{\acronymentry}[1]{%
10340     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10341     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10342 }

```

footnote-sc-desc \textsc{\<short>}\footnote{\<long>} acronym style that has an accompanying description (which the user needs to supply).

```

10343 \renewacronymstyle{footnote-sc-desc}%
10344 {%
10345   \GlsUseAcrEntryDisplayStyle{footnote-sc}%
10346 }%
10347 {%
10348   \GlsUseAcrStyleDefs{footnote-sc}%
10349   \renewcommand*{\GenericAcronymFields}{}%
10350   \renewcommand*{\acronymsort}[2]{##2}%
10351   \renewcommand*{\acronymentry}[1]{%
10352     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10353     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10354 }

```

footnote-sm-desc \textsmaller{\<short>}\footnote{\<long>} acronym style that has an accompanying description (which the user needs to supply).

```

10355 \renewacronymstyle{footnote-sm-desc}%
10356 {%
10357   \GlsUseAcrEntryDisplayStyle{footnote-sm}%
10358 }%
10359 {%
10360   \GlsUseAcrStyleDefs{footnote-sm}%
10361   \renewcommand*{\GenericAcronymFields}{}%
10362   \renewcommand*{\acronymsort}[2]{##2}%
10363   \renewcommand*{\acronymentry}[1]{%
10364     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10365     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10366 }

```

Use \newacronymhook to modify the key list to set the access text to the long version by default.

```

10367 \renewcommand*{\newacronymhook}{%
10368   \edef\@gls@keylist{shortaccess=\the\glslongtok,%
10369     \the\glskeylisttok}%
10370   \expandafter\glskeylisttok\expandafter{\@gls@keylist}%
10371 }

```

defaultNewAcronymDef Modify default style to use access text:

```

10372 \renewcommand*{\DefaultNewAcronymDef}{%
10373   \edef\@do@newglossaryentry{%
10374     \noexpand\newglossaryentry{\the\glslabeltok}%
10375     {%
10376       type=\acronymtype,%
10377       name={\the\glsshorttok},%
10378       description={\the\glslongtok},%
10379       descriptionaccess=\relax,
10380       text={\the\glsshorttok},%
10381       access={\noexpand\@glo@textaccess},%
10382       sort={\the\glsshorttok},%
10383       short={\the\glsshorttok},%
10384       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10385       shortaccess={\the\glslongtok},%
10386       long={\the\glslongtok},%
10387       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10388       descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10389       first={\noexpand\glslongaccessdisplay
10390         {\the\glslongtok}{\the\glslabeltok}\space
10391         (\noexpand\glsshortaccessdisplay
10392         {\the\glsshorttok}{\the\glslabeltok})},%
10393       plural={\the\glsshorttok\acrpluralsuffix},%
10394       firstplural={\noexpand\glslongpluralaccessdisplay
10395         {\noexpand\@glo@longpl}{\the\glslabeltok}\space
10396         (\noexpand\glsshortpluralaccessdisplay
10397         {\noexpand\@glo@shortpl}{\the\glslabeltok})},%
10398       firstaccess=\relax,
10399       firstpluralaccess=\relax,
10400       textaccess={\noexpand\@glo@shortaccess},%
10401       \the\glskeylisttok
10402     }%
10403   }%
10404   \let\@org@gls@assign@firstpl\gls@assign@firstpl
10405   \let\@org@gls@assign@plural\gls@assign@plural
10406   \let\@org@gls@assign@descplural\gls@assign@descplural
10407   \def\gls@assign@firstpl##1##2{%
10408     \@@gls@expand@field{##1}{firstpl}{##2}%
10409   }%
10410   \def\gls@assign@plural##1##2{%
10411     \@@gls@expand@field{##1}{plural}{##2}%
10412   }%
10413   \def\gls@assign@descplural##1##2{%
10414     \@@gls@expand@field{##1}{descplural}{##2}%
10415   }%
10416   \@do@newglossaryentry
10417   \let\gls@assign@firstpl\@org@gls@assign@firstpl
10418   \let\gls@assign@plural\@org@gls@assign@plural
10419   \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
10420 }

```

otnoteNewAcronymDef

```

10421 \renewcommand*{\DescriptionFootnoteNewAcronymDef}{%
10422   \edef\@do@newglossaryentry{%
10423     \noexpand\newglossaryentry{\the\glslabeltok}%
10424     {%
10425       type=\acronymtype,%
10426       name={\noexpand\acronymfont{\the\glsshorttok}},%
10427       sort={\the\glsshorttok},%
10428       text={\the\glsshorttok},%
10429       short={\the\glsshorttok},%
10430       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10431       shortaccess={\the\glslongtok},%
10432       long={\the\glslongtok},%
10433       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10434       access={\noexpand\@glo@textaccess},%
10435       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10436       symbol={\the\glslongtok},%
10437       symbolplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10438       firstpluralaccess=\relax,
10439       textaccess={\noexpand\@glo@shortaccess},%
10440       \the\glskeylisttok
10441     }%
10442   }%
10443   \let\@org@gls@assign@firstpl\gls@assign@firstpl
10444   \let\@org@gls@assign@plural\gls@assign@plural
10445   \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
10446   \def\gls@assign@firstpl##1##2{%
10447     \@@gls@expand@field{##1}{firstpl}{##2}%
10448   }%
10449   \def\gls@assign@plural##1##2{%
10450     \@@gls@expand@field{##1}{plural}{##2}%
10451   }%
10452   \def\gls@assign@symbolplural##1##2{%
10453     \@@gls@expand@field{##1}{symbolplural}{##2}%
10454   }%
10455   \@do@newglossaryentry
10456   \let\gls@assign@plural\@org@gls@assign@plural
10457   \let\gls@assign@firstpl\@org@gls@assign@firstpl
10458   \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
10459 }

```

ipitionNewAcronymDef

```

10460 \renewcommand*{\DescriptionNewAcronymDef}{%
10461   \edef\@do@newglossaryentry{%
10462     \noexpand\newglossaryentry{\the\glslabeltok}%
10463     {%
10464       type=\acronymtype,%
10465       name={\noexpand
10466         \acrnameformat{\the\glsshorttok}{\the\glslongtok}},%

```

```

10467     access={\noexpand\@glo@textaccess},%
10468     sort={\the\glsshorttok},%
10469     short={\the\glsshorttok},%
10470     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10471     shortaccess={\the\glslongtok},%
10472     long={\the\glslongtok},%
10473     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10474     first={\the\glslongtok},%
10475     firstaccess=\relax,
10476     firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10477     text={\the\glsshorttok},%
10478     textaccess={\the\glslongtok},%
10479     plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10480     symbol={\noexpand\@glo@text},%
10481     symbolaccess={\noexpand\@glo@textaccess},%
10482     symbolplural={\noexpand\@glo@plural},%
10483     firstpluralaccess=\relax,
10484     textaccess={\noexpand\@glo@shortaccess},%
10485     \the\glskeylisttok}%
10486 }%
10487 \let\@org@gls@assign@firstpl\gls@assign@firstpl
10488 \let\@org@gls@assign@plural\gls@assign@plural
10489 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
10490 \def\gls@assign@firstpl##1##2{%
10491   \@@gls@expand@field{##1}{firstpl}{##2}%
10492 }%
10493 \def\gls@assign@plural##1##2{%
10494   \@@gls@expand@field{##1}{plural}{##2}%
10495 }%
10496 \def\gls@assign@symbolplural##1##2{%
10497   \@@gls@expand@field{##1}{symbolplural}{##2}%
10498 }%
10499 \do@newglossaryentry
10500 \let\gls@assign@firstpl\@org@gls@assign@firstpl
10501 \let\gls@assign@plural\@org@gls@assign@plural
10502 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
10503 }

```

otnoteNewAcronymDef

```

10504 \renewcommand*{\FootnoteNewAcronymDef}{%
10505   \edef\@do@newglossaryentry{%
10506     \noexpand\newglossaryentry{\the\glslabeltok}%
10507     {%
10508       type=\acronymtype,%
10509       name={\noexpand\acronymfont{\the\glsshorttok}},%
10510       sort={\the\glsshorttok},%
10511       text={\the\glsshorttok},%
10512       textaccess={\the\glslongtok},%
10513       access={\noexpand\@glo@textaccess},%

```

```

10514 plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10515 short={\the\glsshorttok},%
10516 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10517 long={\the\glslongtok},%
10518 longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10519 description={\the\glslongtok},%
10520 descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10521 \the\glskeylisttok
10522 }%
10523 }%
10524 \let\@org@gls@assign@plural\gls@assign@plural
10525 \let\@org@gls@assign@firstpl\gls@assign@firstpl
10526 \let\@org@gls@assign@descplural\gls@assign@descplural
10527 \def\gls@assign@firstpl##1##2{%
10528   \@@gls@expand@field{##1}{firstpl}{##2}%
10529 }%
10530 \def\gls@assign@plural##1##2{%
10531   \@@gls@expand@field{##1}{plural}{##2}%
10532 }%
10533 \def\gls@assign@descplural##1##2{%
10534   \@@gls@expand@field{##1}{descplural}{##2}%
10535 }%
10536 \@do@newglossaryentry
10537 \let\gls@assign@plural\@org@gls@assign@plural
10538 \let\gls@assign@firstpl\@org@gls@assign@firstpl
10539 \let\gls@assign@descplural\@org@gls@assign@descplural
10540 }

```

\SmallNewAcronymDef

```

10541 \renewcommand*{\SmallNewAcronymDef}{%
10542   \edef\@do@newglossaryentry{%
10543     \noexpand\newglossaryentry{\the\glslabeltok}%
10544     {%
10545       type=\acronymtype,%
10546       name={\noexpand\acronymfont{\the\glsshorttok}},%
10547       access={\noexpand\@glo@symbolaccess},%
10548       sort={\the\glsshorttok},%
10549       short={\the\glsshorttok},%
10550       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10551       shortaccess={\the\glslongtok},%
10552       long={\the\glslongtok},%
10553       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10554       text={\noexpand\@glo@short},%
10555       textaccess={\noexpand\@glo@shortaccess},%
10556       plural={\noexpand\@glo@shortpl},%
10557       first={\the\glslongtok},%
10558       firstaccess=\relax,%
10559       firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10560       description={\noexpand\@glo@first},%

```

```

10561     descriptionplural={\noexpand\@glo@firstplural},%
10562     symbol={\the\glsshorttok},%
10563     symbolaccess={\the\glslongtok},%
10564     symbolplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10565     \the\glskeylisttok
10566   }%
10567 }%
10568 \let\@org@gls@assign@firstpl\gls@assign@firstpl
10569 \let\@org@gls@assign@plural\gls@assign@plural
10570 \let\@org@gls@assign@descplural\gls@assign@descplural
10571 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
10572 \def\gls@assign@firstpl##1##2{%
10573   \@@gls@expand@field{##1}{firstpl}{##2}%
10574 }%
10575 \def\gls@assign@plural##1##2{%
10576   \@@gls@expand@field{##1}{plural}{##2}%
10577 }%
10578 \def\gls@assign@descplural##1##2{%
10579   \@@gls@expand@field{##1}{descplural}{##2}%
10580 }%
10581 \def\gls@assign@symbolplural##1##2{%
10582   \@@gls@expand@field{##1}{symbolplural}{##2}%
10583 }%
10584 \do@newglossaryentry
10585 \let\gls@assign@firstpl\@org@gls@assign@firstpl
10586 \let\gls@assign@plural\@org@gls@assign@plural
10587 \let\gls@assign@descplural\@org@gls@assign@descplural
10588 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
10589 }

```

The following are kept for compatibility with versions before 3.0:

```

\glsshortaccesskey
10590   \newcommand*{\glsshortaccesskey}{\glsshortkey access}%

hortpluralaccesskey
10591   \newcommand*{\glsshortpluralaccesskey}{\glsshortpluralkey access}%

\glslongaccesskey
10592   \newcommand*{\glslongaccesskey}{\glslongkey access}%

longpluralaccesskey
10593   \newcommand*{\glslongpluralaccesskey}{\glslongpluralkey access}%

```

7.5 Debugging Commands

```

\showglongnameaccess
10594 \newcommand*{\showglongnameaccess}[1]{%
10595   \expandafter\show\curname glo@\glsdetoklabel{#1}@textaccess\endcsname
10596 }

```

```

\showglotextaccess
10597 \newcommand*{\showglotextaccess}[1]{%
10598   \expandafter\show\csname glo@\glsdetoklabel{#1}@textaccess\endcsname
10599 }

showglopluralaccess
10600 \newcommand*{\showglopluralaccess}[1]{%
10601   \expandafter\show\csname glo@\glsdetoklabel{#1}@pluralaccess\endcsname
10602 }

\showglofirstaccess
10603 \newcommand*{\showglofirstaccess}[1]{%
10604   \expandafter\show\csname glo@\glsdetoklabel{#1}@firstaccess\endcsname
10605 }

lofirstpluralaccess
10606 \newcommand*{\showglofirstpluralaccess}[1]{%
10607   \expandafter\show\csname glo@\glsdetoklabel{#1}@firstpluralaccess\endcsname
10608 }

showglosymbolaccess
10609 \newcommand*{\showglosymbolaccess}[1]{%
10610   \expandafter\show\csname glo@\glsdetoklabel{#1}@symbolaccess\endcsname
10611 }

osymbolpluralaccess
10612 \newcommand*{\showglosymbolpluralaccess}[1]{%
10613   \expandafter\show\csname glo@\glsdetoklabel{#1}@symbolpluralaccess\endcsname
10614 }

\showglodescaccess
10615 \newcommand*{\showglodescaccess}[1]{%
10616   \expandafter\show\csname glo@\glsdetoklabel{#1}@descaccess\endcsname
10617 }

glodescpluralaccess
10618 \newcommand*{\showglodescpluralaccess}[1]{%
10619   \expandafter\show\csname glo@\glsdetoklabel{#1}@descpluralaccess\endcsname
10620 }

\showgloshortaccess
10621 \newcommand*{\showgloshortaccess}[1]{%
10622   \expandafter\show\csname glo@\glsdetoklabel{#1}@shortaccess\endcsname
10623 }

loshortpluralaccess
10624 \newcommand*{\showgloshortpluralaccess}[1]{%
10625   \expandafter\show\csname glo@\glsdetoklabel{#1}@shortpluralaccess\endcsname
10626 }

```

```

\showglolongaccess
10627 \newcommand*{\showglolongaccess}[1]{%
10628   \expandafter\show\csname glo@\glsdetoklabel{#1}@longaccess\endcsname
10629 }

glolongpluralaccess
10630 \newcommand*{\showglolongpluralaccess}[1]{%
10631   \expandafter\show\csname glo@\glsdetoklabel{#1}@longpluralaccess\endcsname
10632 }

```

8 Multi-Lingual Support

Many thanks to everyone who contributed to the translations both via email and on comp.text.tex.

8.1 Babel Captions

Define captions if multi-lingual support is required, but the package is not loaded.

```

10633 \NeedsTeXFormat{LaTeX2e}
10634 \ProvidesPackage{glossaries-babel}[2013/11/14 v4.0 (NLCT)]

```

English:

```

10635 \@ifundefined{captionsenglish}{-}{%
10636   \addto\captionsenglish{%
10637     \renewcommand*{\glossaryname}{Glossary}%
10638     \renewcommand*{\acronymname}{Acronyms}%
10639     \renewcommand*{\entryname}{Notation}%
10640     \renewcommand*{\descriptionname}{Description}%
10641     \renewcommand*{\symbolname}{Symbol}%
10642     \renewcommand*{\pagelistname}{Page List}%
10643     \renewcommand*{\glssymbolsgroupname}{Symbols}%
10644     \renewcommand*{\glsnumbersgroupname}{Numbers}%
10645 }%
10646 }

10647 \@ifundefined{captionsamerican}{-}{%
10648   \addto\captionsamerican{%
10649     \renewcommand*{\glossaryname}{Glossary}%
10650     \renewcommand*{\acronymname}{Acronyms}%
10651     \renewcommand*{\entryname}{Notation}%
10652     \renewcommand*{\descriptionname}{Description}%
10653     \renewcommand*{\symbolname}{Symbol}%
10654     \renewcommand*{\pagelistname}{Page List}%
10655     \renewcommand*{\glssymbolsgroupname}{Symbols}%
10656     \renewcommand*{\glsnumbersgroupname}{Numbers}%
10657 }%
10658 }

10659 \@ifundefined{captionsaustralian}{-}{%

```



```

10660 \addto\captionsaustrian{%
10661     \renewcommand*{\glossaryname}{Glossary}%
10662     \renewcommand*{\acronymname}{Acronyms}%
10663     \renewcommand*{\entryname}{Notation}%
10664     \renewcommand*{\descriptionname}{Description}%
10665     \renewcommand*{\symbolname}{Symbol}%
10666     \renewcommand*{\pagelistname}{Page List}%
10667     \renewcommand*{\glssymbolsgroupname}{Symbols}%
10668     \renewcommand*{\glsnumbersgroupname}{Numbers}%
10669 }%
10670 }
10671 \@ifundefined{captionsbritish}{-}{%
10672     \addto\captionsbritish{%
10673         \renewcommand*{\glossaryname}{Glossary}%
10674         \renewcommand*{\acronymname}{Acronyms}%
10675         \renewcommand*{\entryname}{Notation}%
10676         \renewcommand*{\descriptionname}{Description}%
10677         \renewcommand*{\symbolname}{Symbol}%
10678         \renewcommand*{\pagelistname}{Page List}%
10679         \renewcommand*{\glssymbolsgroupname}{Symbols}%
10680         \renewcommand*{\glsnumbersgroupname}{Numbers}%
10681     }%
10682 \@ifundefined{captionscanadian}{-}{%
10683     \addto\captionscanadian{%
10684         \renewcommand*{\glossaryname}{Glossary}%
10685         \renewcommand*{\acronymname}{Acronyms}%
10686         \renewcommand*{\entryname}{Notation}%
10687         \renewcommand*{\descriptionname}{Description}%
10688         \renewcommand*{\symbolname}{Symbol}%
10689         \renewcommand*{\pagelistname}{Page List}%
10690         \renewcommand*{\glssymbolsgroupname}{Symbols}%
10691         \renewcommand*{\glsnumbersgroupname}{Numbers}%
10692     }%
10693 }
10694 \@ifundefined{captionsnewzealand}{-}{%
10695     \addto\captionnewzealand{%
10696         \renewcommand*{\glossaryname}{Glossary}%
10697         \renewcommand*{\acronymname}{Acronyms}%
10698         \renewcommand*{\entryname}{Notation}%
10699         \renewcommand*{\descriptionname}{Description}%
10700         \renewcommand*{\symbolname}{Symbol}%
10701         \renewcommand*{\pagelistname}{Page List}%
10702         \renewcommand*{\glssymbolsgroupname}{Symbols}%
10703         \renewcommand*{\glsnumbersgroupname}{Numbers}%
10704     }%
10705 }
10706 \@ifundefined{captionsUKenglish}{-}{%
10707     \addto\captionUKenglish{%
10708         \renewcommand*{\glossaryname}{Glossary}%

```

```

10709 \renewcommand*{\acronymname}{Acronyms}%
10710 \renewcommand*{\entryname}{Notation}%
10711 \renewcommand*{\descriptionname}{Description}%
10712 \renewcommand*{\symbolname}{Symbol}%
10713 \renewcommand*{\pagelistname}{Page List}%
10714 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10715 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10716 }%
10717 }
10718 \@ifundefined{captionsUSenglish}{-}{%
10719 \addto\captionsUSenglish{%
10720 \renewcommand*{\glossaryname}{Glossary}%
10721 \renewcommand*{\acronymname}{Acronyms}%
10722 \renewcommand*{\entryname}{Notation}%
10723 \renewcommand*{\descriptionname}{Description}%
10724 \renewcommand*{\symbolname}{Symbol}%
10725 \renewcommand*{\pagelistname}{Page List}%
10726 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10727 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10728 }%
10729 }

```

German (quite a few variations were suggested for German; I settled on the following):

```

10730 \@ifundefined{captionsgerman}{-}{%
10731 \addto\captionsgerman{%
10732 \renewcommand*{\glossaryname}{Glossar}%
10733 \renewcommand*{\acronymname}{Akronyme}%
10734 \renewcommand*{\entryname}{Bezeichnung}%
10735 \renewcommand*{\descriptionname}{Beschreibung}%
10736 \renewcommand*{\symbolname}{Symbol}%
10737 \renewcommand*{\pagelistname}{Seiten}%
10738 \renewcommand*{\glssymbolsgroupname}{Symbole}%
10739 \renewcommand*{\glsnumbersgroupname}{Zahlen}}
10740 }

```

ngerman is identical to German:

```

10741 \@ifundefined{captionsgerman}{-}{%
10742 \addto\captionsgerman{%
10743 \renewcommand*{\glossaryname}{Glossar}%
10744 \renewcommand*{\acronymname}{Akronyme}%
10745 \renewcommand*{\entryname}{Bezeichnung}%
10746 \renewcommand*{\descriptionname}{Beschreibung}%
10747 \renewcommand*{\symbolname}{Symbol}%
10748 \renewcommand*{\pagelistname}{Seiten}%
10749 \renewcommand*{\glssymbolsgroupname}{Symbole}%
10750 \renewcommand*{\glsnumbersgroupname}{Zahlen}}
10751 }

```

Italian:

```

10752 \@ifundefined{captionsitalian}{-}{%

```

```

10753 \addto\captionssitalian{%
10754   \renewcommand*{\glossaryname}{Glossario}%
10755   \renewcommand*{\acronymname}{Acronimi}%
10756   \renewcommand*{\entryname}{Nomenclatura}%
10757   \renewcommand*{\descriptionname}{Descrizione}%
10758   \renewcommand*{\symbolname}{Simbolo}%
10759   \renewcommand*{\pagelistname}{Elenco delle pagine}%
10760   \renewcommand*{\glssymbolsgroupname}{Simboli}%
10761   \renewcommand*{\glsnumbersgroupname}{Numeri}}
10762 }

```

Dutch:

```

10763 \@ifundefined{captionssdutch}{}{%
10764   \addto\captionssdutch{%
10765     \renewcommand*{\glossaryname}{Woordenlijst}%
10766     \renewcommand*{\acronymname}{Acroniemen}%
10767     \renewcommand*{\entryname}{Benaming}%
10768     \renewcommand*{\descriptionname}{Beschrijving}%
10769     \renewcommand*{\symbolname}{Symbool}%
10770     \renewcommand*{\pagelistname}{Pagina's}%
10771     \renewcommand*{\glssymbolsgroupname}{Symbolen}%
10772     \renewcommand*{\glsnumbersgroupname}{Cijfers}}
10773 }

```

Spanish:

```

10774 \@ifundefined{captionssspanish}{}{%
10775   \addto\captionssspanish{%
10776     \renewcommand*{\glossaryname}{Glosario}%
10777     \renewcommand*{\acronymname}{Siglas}%
10778     \renewcommand*{\entryname}{Entrada}%
10779     \renewcommand*{\descriptionname}{Descripci\'on}%
10780     \renewcommand*{\symbolname}{S\'imbolo}%
10781     \renewcommand*{\pagelistname}{Lista de p\'aginas}%
10782     \renewcommand*{\glssymbolsgroupname}{S\'imbolos}%
10783     \renewcommand*{\glsnumbersgroupname}{N\'umeros}}
10784 }

```

French:

```

10785 \@ifundefined{captionssfrench}{}{%
10786   \addto\captionssfrench{%
10787     \renewcommand*{\glossaryname}{Glossaire}%
10788     \renewcommand*{\acronymname}{Acronymes}%
10789     \renewcommand*{\entryname}{Termes}%
10790     \renewcommand*{\descriptionname}{Description}%
10791     \renewcommand*{\symbolname}{Symbole}%
10792     \renewcommand*{\pagelistname}{Pages}%
10793     \renewcommand*{\glssymbolsgroupname}{Symboles}%
10794     \renewcommand*{\glsnumbersgroupname}{Nombres}}
10795 }
10796 \@ifundefined{captionssfrenchb}{}{%
10797   \addto\captionssfrenchb{%

```

```

10798 \renewcommand*{\glossaryname}{Glossaire}%
10799 \renewcommand*{\acronymname}{Acronymes}%
10800 \renewcommand*{\entryname}{Terme}%
10801 \renewcommand*{\descriptionname}{Description}%
10802 \renewcommand*{\symbolname}{Symbole}%
10803 \renewcommand*{\pagelistname}{Pages}%
10804 \renewcommand*{\glssymbolsgroupname}{Symboles}%
10805 \renewcommand*{\glsnumbersgroupname}{Nombres}}
10806 }
10807 \@ifundefined{captionsfrancais}{}{%
10808 \addto\captionsfrancais{%
10809 \renewcommand*{\glossaryname}{Glossaire}%
10810 \renewcommand*{\acronymname}{Acronymes}%
10811 \renewcommand*{\entryname}{Terme}%
10812 \renewcommand*{\descriptionname}{Description}%
10813 \renewcommand*{\symbolname}{Symbole}%
10814 \renewcommand*{\pagelistname}{Pages}%
10815 \renewcommand*{\glssymbolsgroupname}{Symboles}%
10816 \renewcommand*{\glsnumbersgroupname}{Nombres}}
10817 }

```

Danish:

```

10818 \@ifundefined{captionsdanish}{}{%
10819 \addto\captionsdanish{%
10820 \renewcommand*{\glossaryname}{Ordliste}%
10821 \renewcommand*{\acronymname}{Akronymer}%
10822 \renewcommand*{\entryname}{Symbolforklaring}%
10823 \renewcommand*{\descriptionname}{Beskrivelse}%
10824 \renewcommand*{\symbolname}{Symbol}%
10825 \renewcommand*{\pagelistname}{Side}%
10826 \renewcommand*{\glssymbolsgroupname}{Symboler}%
10827 \renewcommand*{\glsnumbersgroupname}{Tal}}
10828 }

```

Irish:

```

10829 \@ifundefined{captionsirish}{}{%
10830 \addto\captionsirish{%
10831 \renewcommand*{\glossaryname}{Gluais}%
10832 \renewcommand*{\acronymname}{Acrainmneacha}%

```

wasn't sure whether to go for Nóta (Note), Ciall ('Meaning', 'sense') or Brí ('Meaning'). In the end I chose Ciall.

```

10833 \renewcommand*{\entryname}{Ciall}%
10834 \renewcommand*{\descriptionname}{Tuairisc}%

```

Again, not sure whether to use Comhartha/Comharthaí or Siombail/Siombaile, so have chosen the former.

```

10835 \renewcommand*{\symbolname}{Comhartha}%
10836 \renewcommand*{\glssymbolsgroupname}{Comhartha'\{i}}%
10837 \renewcommand*{\pagelistname}{Leathanaigh}%
10838 \renewcommand*{\glsnumbersgroupname}{Uimhreacha}}

```

10839 }

Hungarian:

```
10840 \@ifundefined{captionsmagyar}{%{}%
10841   \addto\captionsmagyar{%
10842     \renewcommand*{\glossaryname}{Sz\`ojegyz\`ek}%
10843     \renewcommand*{\acronymname}{Bet\H uszavak}%
10844     \renewcommand*{\entryname}{Kifejez\`es}%
10845     \renewcommand*{\descriptionname}{Magyar\`azat}%
10846     \renewcommand*{\symbolname}{Jel\"ol\`es}%
10847     \renewcommand*{\pagelistname}{Oldalsz\`am}%
10848     \renewcommand*{\glssymbolsgroupname}{Jelek}%
10849     \renewcommand*{\glsnumbersgroupname}{Sz\`amjegyek}%
10850   }
10851 }
10852 \@ifundefined{captionshungarian}{%{}%
10853   \addto\captionshungarian{%
10854     \renewcommand*{\glossaryname}{Sz\`ojegyz\`ek}%
10855     \renewcommand*{\acronymname}{Bet\H uszavak}%
10856     \renewcommand*{\entryname}{Kifejez\`es}%
10857     \renewcommand*{\descriptionname}{Magyar\`azat}%
10858     \renewcommand*{\symbolname}{Jel\"ol\`es}%
10859     \renewcommand*{\pagelistname}{Oldalsz\`am}%
10860     \renewcommand*{\glssymbolsgroupname}{Jelek}%
10861     \renewcommand*{\glsnumbersgroupname}{Sz\`amjegyek}%
10862   }
10863 }
```

Polish

```
10864 \@ifundefined{captionspolish}{%{}%
10865   \addto\captionspolish{%
10866     \renewcommand*{\glossaryname}{S{\l}ownik termin\`ow}%
10867     \renewcommand*{\acronymname}{Skr\`ot}%
10868     \renewcommand*{\entryname}{Termin}%
10869     \renewcommand*{\descriptionname}{Opis}%
10870     \renewcommand*{\symbolname}{Symbol}%
10871     \renewcommand*{\pagelistname}{Strony}%
10872     \renewcommand*{\glssymbolsgroupname}{Symbole}%
10873     \renewcommand*{\glsnumbersgroupname}{Liczby}}
10874 }
```

Brazilian

```
10875 \@ifundefined{captionsbrazil}{%{}%
10876   \addto\captionsbrazil{%
10877     \renewcommand*{\glossaryname}{Gloss\`ario}%
10878     \renewcommand*{\acronymname}{Siglas}%
10879     \renewcommand*{\entryname}{Nota\c c\~ao}%
10880     \renewcommand*{\descriptionname}{Descri\c c\~ao}%
10881     \renewcommand*{\symbolname}{S\`imbolo}%
10882     \renewcommand*{\pagelistname}{Lista de P\`aginas}%
10883     \renewcommand*{\glssymbolsgroupname}{S\`imbolos}%

```

```

10884 \renewcommand*{\glsnumbersgroupname}{N\,umeros}%
10885 }%
10886 }

```

8.2 Polyglossia Captions

```

10887 \NeedsTeXFormat{LaTeX2e}
10888 \ProvidesPackage{glossaries-polyglossia}[2013/11/14 v4.0 (NLCT)]

```

English:

```

10889 \@ifundefined{captionsenglish}{}{%
10890 \expandafter\toks@\expandafter{\captionsenglish
10891 \renewcommand*{\glossaryname}{\textenglish{Glossary}}%
10892 \renewcommand*{\acronymname}{\textenglish{Acronyms}}%
10893 \renewcommand*{\entryname}{\textenglish{Notation}}%
10894 \renewcommand*{\descriptionname}{\textenglish{Description}}%
10895 \renewcommand*{\symbolname}{\textenglish{Symbol}}%
10896 \renewcommand*{\pagelistname}{\textenglish{Page List}}%
10897 \renewcommand*{\glssymbolsgroupname}{\textenglish{Symbols}}%
10898 \renewcommand*{\glsnumbersgroupname}{\textenglish{Numbers}}%
10899 }%
10900 \edef\captionsenglish{\the\toks@}%
10901 }

```

German:

```

10902 \@ifundefined{captionsgerman}{}{%
10903 \expandafter\toks@\expandafter{\captionsgerman
10904 \renewcommand*{\glossaryname}{\textgerman{Glossar}}%
10905 \renewcommand*{\acronymname}{\textgerman{Akronyme}}%
10906 \renewcommand*{\entryname}{\textgerman{Bezeichnung}}%
10907 \renewcommand*{\descriptionname}{\textgerman{Beschreibung}}%
10908 \renewcommand*{\symbolname}{\textgerman{Symbol}}%
10909 \renewcommand*{\pagelistname}{\textgerman{Seiten}}%
10910 \renewcommand*{\glssymbolsgroupname}{\textgerman{Symbole}}%
10911 \renewcommand*{\glsnumbersgroupname}{\textgerman{Zahlen}}%
10912 }%
10913 \edef\captionsgerman{\the\toks@}%
10914 }

```

Italian:

```

10915 \@ifundefined{captionsitalian}{}{%
10916 \expandafter\toks@\expandafter{\captionsitalian
10917 \renewcommand*{\glossaryname}{\textitalian{Glossario}}%
10918 \renewcommand*{\acronymname}{\textitalian{Acronimi}}%
10919 \renewcommand*{\entryname}{\textitalian{Nomenclatura}}%
10920 \renewcommand*{\descriptionname}{\textitalian{Descrizione}}%
10921 \renewcommand*{\symbolname}{\textitalian{Simbolo}}%
10922 \renewcommand*{\pagelistname}{\textitalian{Elenco delle pagine}}%
10923 \renewcommand*{\glssymbolsgroupname}{\textitalian{Simboli}}%
10924 \renewcommand*{\glsnumbersgroupname}{\textitalian{Numeri}}%
10925 }%

```

```

10926 \edef\captionssitalian{\the\toks@}%
10927 }

```

Dutch:

```

10928 \@ifundefined{captionsdutch}{\{}%
10929 \expandafter\toks@\expandafter{\captionsdutch
10930 \renewcommand*{\glossaryname}{\textdutch{Woordenlijst}}%
10931 \renewcommand*{\acronymname}{\textdutch{Acroniemen}}%
10932 \renewcommand*{\entryname}{\textdutch{Benaming}}%
10933 \renewcommand*{\descriptionname}{\textdutch{Beschrijving}}%
10934 \renewcommand*{\symbolname}{\textdutch{Symbool}}%
10935 \renewcommand*{\pagelistname}{\textdutch{Pagina's}}%
10936 \renewcommand*{\glssymbolsgroupname}{\textdutch{Symbolen}}%
10937 \renewcommand*{\glsnumbersgroupname}{\textdutch{Cijfers}}%
10938 }%
10939 \edef\captionsdutch{\the\toks@}%
10940 }

```

Spanish:

```

10941 \@ifundefined{captionsspanish}{\{}%
10942 \expandafter\toks@\expandafter{\captionsspanish
10943 \renewcommand*{\glossaryname}{\textspanish{Glosario}}%
10944 \renewcommand*{\acronymname}{\textspanish{Siglas}}%
10945 \renewcommand*{\entryname}{\textspanish{Entrada}}%
10946 \renewcommand*{\descriptionname}{\textspanish{Descripci'on}}%
10947 \renewcommand*{\symbolname}{\textspanish{S'\{i}mbolo}}%
10948 \renewcommand*{\pagelistname}{\textspanish{Lista de p'aginas}}%
10949 \renewcommand*{\glssymbolsgroupname}{\textspanish{S'\{i}mbolos}}%
10950 \renewcommand*{\glsnumbersgroupname}{\textspanish{N'umeros}}%
10951 }%
10952 \edef\captionsspanish{\the\toks@}%
10953 }

```

French:

```

10954 \@ifundefined{captionsfrench}{\{}%
10955 \expandafter\toks@\expandafter{\captionsfrench
10956 \renewcommand*{\glossaryname}{\textfrench{Glossaire}}%
10957 \renewcommand*{\acronymname}{\textfrench{Acronymes}}%
10958 \renewcommand*{\entryname}{\textfrench{Terme}}%
10959 \renewcommand*{\descriptionname}{\textfrench{Description}}%
10960 \renewcommand*{\symbolname}{\textfrench{Symbole}}%
10961 \renewcommand*{\pagelistname}{\textfrench{Pages}}%
10962 \renewcommand*{\glssymbolsgroupname}{\textfrench{Symboles}}%
10963 \renewcommand*{\glsnumbersgroupname}{\textfrench{Nombres}}%
10964 }%
10965 \edef\captionsfrench{\the\toks@}%
10966 }

```

Danish:

```

10967 \@ifundefined{captionsdanish}{\{}%
10968 \expandafter\toks@\expandafter{\captionsdanish
10969 \renewcommand*{\glossaryname}{\textdanish{Ordliste}}%

```

```

10970 \renewcommand*{\acronymname}{\textdanish{Akronymer}}}%
10971 \renewcommand*{\entryname}{\textdanish{Symbolforklaring}}}%
10972 \renewcommand*{\descriptionname}{\textdanish{Beskrivelse}}}%
10973 \renewcommand*{\symbolname}{\textdanish{Symbol}}}%
10974 \renewcommand*{\pagelistname}{\textdanish{Side}}}%
10975 \renewcommand*{\glssymbolsgroupname}{\textdanish{Symboler}}}%
10976 \renewcommand*{\glslnumbersgroupname}{\textdanish{Tal}}}%
10977 }%
10978 \edef\captionsdanish{\the\toks@}%
10979 }

Irish:
10980 \@ifundefined{captionsirish}{}{%
10981 \expandafter\toks@\expandafter{\captionsirish
10982 \renewcommand*{\glossaryname}{\textirish{Gluais}}}%
10983 \renewcommand*{\acronymname}{\textirish{Acrainmneacha}}}%
10984 \renewcommand*{\entryname}{\textirish{Ciall}}}%
10985 \renewcommand*{\descriptionname}{\textirish{Tuirisc}}}%
10986 \renewcommand*{\symbolname}{\textirish{Comhartha}}}%
10987 \renewcommand*{\glssymbolsgroupname}{\textirish{Comhartha\,'{i}}}}%
10988 \renewcommand*{\pagelistname}{\textirish{Leathanaigh}}}%
10989 \renewcommand*{\glslnumbersgroupname}{\textirish{Uimhreacha}}}%
10990 }%
10991 \edef\captionsirish{\the\toks@}%
10992 }

Hungarian:
10993 \@ifundefined{captionsmagyar}{}{%
10994 \expandafter\toks@\expandafter{\captionsmagyar
10995 \renewcommand*{\glossaryname}{\textmagyar{Sz\,'ojegyz\,'ek}}}%
10996 \renewcommand*{\acronymname}{\textmagyar{Bet\H uszavak}}}%
10997 \renewcommand*{\entryname}{\textmagyar{Kifejez\,'es}}}%
10998 \renewcommand*{\descriptionname}{\textmagyar{Magyar\,'azat}}}%
10999 \renewcommand*{\symbolname}{\textmagyar{Jel\,"ol\,'es}}}%
11000 \renewcommand*{\pagelistname}{\textmagyar{Oldalsz\,'am}}}%
11001 \renewcommand*{\glssymbolsgroupname}{\textmagyar{Jelek}}}%
11002 \renewcommand*{\glslnumbersgroupname}{\textmagyar{Sz\,'amjegyek}}}%
11003 }%
11004 \edef\captionsmagyar{\the\toks@}%
11005 }

Polish
11006 \@ifundefined{captionspolish}{}{%
11007 \expandafter\toks@\expandafter{\captionspolish
11008 \renewcommand*{\glossaryname}{\textpolish{S{\l}ownik termin\,'ow}}}%
11009 \renewcommand*{\acronymname}{\textpolish{Skr\,'ot}}}%
11010 \renewcommand*{\entryname}{\textpolish{Termin}}}%
11011 \renewcommand*{\descriptionname}{\textpolish{Opis}}}%
11012 \renewcommand*{\symbolname}{\textpolish{Symbol}}}%
11013 \renewcommand*{\pagelistname}{\textpolish{Strony}}}%
11014 \renewcommand*{\glssymbolsgroupname}{\textpolish{Symbole}}}%

```



```

11015 \renewcommand*{\glsnumbersgroupname}{\textpolish{Liczby}}%
11016 }%
11017 \edef\captionspolish{\the\toks@}%
11018 }

```

Portugues

```

11019 \@ifundefined{captionsportuges}{}{%
11020 \expandafter\toks@\expandafter{\captionsportuges
11021 \renewcommand*{\glossaryname}{\textportuges{Gloss\'ario}}%
11022 \renewcommand*{\acronymname}{\textportuges{Siglas}}%
11023 \renewcommand*{\entryname}{\textportuges{Nota\c c\~ao}}%
11024 \renewcommand*{\descriptionname}{\textportuges{Descri\c c\~ao}}%
11025 \renewcommand*{\symbolname}{\textportuges{S\'imbolo}}%
11026 \renewcommand*{\pagelistname}{\textportuges{Lista de P\'aginas}}%
11027 \renewcommand*{\glssymbolsgroupname}{\textportuges{S\'imbolos}}%
11028 \renewcommand*{\glsnumbersgroupname}{\textportuges{N\'umeros}}%
11029 }%
11030 \edef\captionsportuges{\the\toks@}%
11031 }

```

8.3 Brazilian Dictionary

This is a dictionary file provided by Thiago de Melo for use with the package.

```

11032 \ProvidesDictionary{glossaries-dictionary}{Brazilian}

```

Provide Brazilian translations:

```

11033 \providetranslation{Glossary}{Gloss\'ario}
11034 \providetranslation{Acronyms}{Siglas}
11035 \providetranslation{Notation (glossaries)}{Nota\c c\~ao}
11036 \providetranslation{Description (glossaries)}{Descri\c c\~ao}
11037 \providetranslation{Symbol (glossaries)}{S\'imbolo}
11038 \providetranslation{Page List (glossaries)}{Lista de P\'aginas}
11039 \providetranslation{Symbols (glossaries)}{S\'imbolos}
11040 \providetranslation{Numbers (glossaries)}{N\'umeros}

```

8.4 Danish Dictionary

This is a dictionary file provided for use with the package.

```

11041 \ProvidesDictionary{glossaries-dictionary}{Danish}

```

Provide Danish translations:

```

11042 \providetranslation{Glossary}{Ordliste}
11043 \providetranslation{Acronyms}{Akronymer}
11044 \providetranslation{Notation (glossaries)}{Symbolforklaring}
11045 \providetranslation{Description (glossaries)}{Beskrivelse}
11046 \providetranslation{Symbol (glossaries)}{Symbol}
11047 \providetranslation{Page List (glossaries)}{Side}
11048 \providetranslation{Symbols (glossaries)}{Symboler}
11049 \providetranslation{Numbers (glossaries)}{Tal}

```

8.5 Dutch Dictionary

This is a dictionary file provided for use with the package.

```
11050 \ProvidesDictionary{glossaries-dictionary}{Dutch}
```

Provide Dutch translations:

```
11051 \providetranslation{Glossary}{Woordenlijst}
11052 \providetranslation{Acronyms}{Acroniemen}
11053 \providetranslation{Notation (glossaries)}{Benaming}
11054 \providetranslation{Description (glossaries)}{Beschrijving}
11055 \providetranslation{Symbol (glossaries)}{Symbool}
11056 \providetranslation{Page List (glossaries)}{Pagina's}
11057 \providetranslation{Symbols (glossaries)}{Symbolen}
11058 \providetranslation{Numbers (glossaries)}{Cijfers}
```

8.6 English Dictionary

This is a dictionary file provided for use with the package.

```
11059 \ProvidesDictionary{glossaries-dictionary}{English}
```

Provide English translations:

```
11060 \providetranslation{Glossary}{Glossary}
11061 \providetranslation{Acronyms}{Acronyms}
11062 \providetranslation{Notation (glossaries)}{Notation}
11063 \providetranslation{Description (glossaries)}{Description}
11064 \providetranslation{Symbol (glossaries)}{Symbol}
11065 \providetranslation{Page List (glossaries)}{Page List}
11066 \providetranslation{Symbols (glossaries)}{Symbols}
11067 \providetranslation{Numbers (glossaries)}{Numbers}
```

8.7 French Dictionary

This is a dictionary file provided for use with the package.

```
11068 \ProvidesDictionary{glossaries-dictionary}{French}
```

Provide French translations:

```
11069 \providetranslation{Glossary}{Glossaire}
11070 \providetranslation{Acronyms}{Acronymes}
11071 \providetranslation{Notation (glossaries)}{Terme}
11072 \providetranslation{Description (glossaries)}{Description}
11073 \providetranslation{Symbol (glossaries)}{Symbole}
11074 \providetranslation{Page List (glossaries)}{Pages}
11075 \providetranslation{Symbols (glossaries)}{Symboles}
11076 \providetranslation{Numbers (glossaries)}{Nombres}
```

8.8 German Dictionary

This is a dictionary file provided for use with the package.

```
11077 \ProvidesDictionary{glossaries-dictionary}{German}
```

Provide German translations (quite a few variations were suggested for German; I settled on the following):

```
11078 \providetranslation{Glossary}{Glossar}
11079 \providetranslation{Acronyms}{Akronyme}
11080 \providetranslation{Notation (glossaries)}{Bezeichnung}
11081 \providetranslation{Description (glossaries)}{Beschreibung}
11082 \providetranslation{Symbol (glossaries)}{Symbol}
11083 \providetranslation{Page List (glossaries)}{Seiten}
11084 \providetranslation{Symbols (glossaries)}{Symbole}
11085 \providetranslation{Numbers (glossaries)}{Zahlen}
```

8.9 Irish Dictionary

This is a dictionary file provided for use with the package.

```
11086 \ProvidesDictionary{glossaries-dictionary}{Irish}
```

Provide Irish translations:

```
11087 \providetranslation{Glossary}{Gluais}
11088 \providetranslation{Acronyms}{Acrainmneacha}
11089 \providetranslation{Notation (glossaries)}{Ciall}
11090 \providetranslation{Description (glossaries)}{Tuairisc}
11091 \providetranslation{Symbol (glossaries)}{Comhartha}
11092 \providetranslation{Page List (glossaries)}{Leathanaigh}
11093 \providetranslation{Symbols (glossaries)}{Comhartha'\{i}}
11094 \providetranslation{Numbers (glossaries)}{Uimhreacha}
```

8.10 Italian Dictionary

This is a dictionary file provided for use with the package.

```
11095 \ProvidesDictionary{glossaries-dictionary}{Italian}
```

Provide Italian translations:

```
11096 \providetranslation{Glossary}{Glossario}
11097 \providetranslation{Acronyms}{Acronimi}
11098 \providetranslation{Notation (glossaries)}{Nomenclatura}
11099 \providetranslation{Description (glossaries)}{Descrizione}
11100 \providetranslation{Symbol (glossaries)}{Simbolo}
11101 \providetranslation{Page List (glossaries)}{Elenco delle pagine}
11102 \providetranslation{Symbols (glossaries)}{Simboli}
11103 \providetranslation{Numbers (glossaries)}{Numeri}
```

8.11 Magyar Dictionary

This is a dictionary file provided for use with the package.

```
11104 \ProvidesDictionary{glossaries-dictionary}{Magyar}
```

Provide translations:

```
11105 \providetranslation{Glossary}{Sz\'ojegy\'ek}
11106 \providetranslation{Acronyms}{Bet\'H uszavak}
```

```

11107 \providetranslation{Notation (glossaries)}{Kifejez\`es}
11108 \providetranslation{Description (glossaries)}{Magyar\`azat}
11109 \providetranslation{Symbol (glossaries)}{Jel\`ol\`es}
11110 \providetranslation{Page List (glossaries)}{Oldalsz\`am}
11111 \providetranslation{Symbols (glossaries)}{Jelek}
11112 \providetranslation{Numbers (glossaries)}{Sz\`amjegyek}

```

8.12 Polish Dictionary

This is a dictionary file provided for use with the package.

```

11113 \ProvidesDictionary{glossaries-dictionary}{Polish}

```

Provide Polish translations:

```

11114 \providetranslation{Glossary}{S{\l}ownik termin\`ow}
11115 \providetranslation{Acronyms}{Skr\`ot}
11116 \providetranslation{Notation (glossaries)}{Termin}
11117 \providetranslation{Description (glossaries)}{Opis}
11118 \providetranslation{Symbol (glossaries)}{Symbol}
11119 \providetranslation{Page List (glossaries)}{Strony}
11120 \providetranslation{Symbols (glossaries)}{Symbole}
11121 \providetranslation{Numbers (glossaries)}{Liczby}

```

8.13 Serbian Dictionary

This dictionary was provided by Zoran Filipovic.

```

11122 \ProvidesDictionary{glossaries-dictionary}{Serbian}
11123 \providetranslation{Glossary}{Mali re\`v cnik}
11124 \providetranslation{Acronyms}{Skra\` cenice}
11125 \providetranslation{Notation (glossaries)}{Oznaka}
11126 \providetranslation{Description (glossaries)}{Opis}
11127 \providetranslation{Symbol (glossaries)}{Simbol}
11128 \providetranslation{Page List (glossaries)}{Stranica}
11129 \providetranslation{Symbols (glossaries)}{Simboli}
11130 \providetranslation{Numbers (glossaries)}{Brojevi}

```

8.14 Spanish Dictionary

This is a dictionary file provided for use with the package.

```

11131 \ProvidesDictionary{glossaries-dictionary}{Spanish}

```

Provide Spanish translations:

```

11132 \providetranslation{Glossary}{Glosario}
11133 \providetranslation{Acronyms}{Siglas}
11134 \providetranslation{Notation (glossaries)}{Entrada}
11135 \providetranslation{Description (glossaries)}{Descripci\`on}
11136 \providetranslation{Symbol (glossaries)}{S\`{\i}mbolo}
11137 \providetranslation{Page List (glossaries)}{Lista de p\`aginas}
11138 \providetranslation{Symbols (glossaries)}{S\`{\i}mbolos}
11139 \providetranslation{Numbers (glossaries)}{N\`umeros}

```

Glossary

`makeindex` An indexing application. 10, 23, 24

`xindy` An flexible indexing application with multilingual support written in Perl. 10, 23, 24

Change History

??		<code>\capitalisewords</code> : made robust	
	super: fixed typo in <code>\subglossentry</code>	242
	(<code>\glossentrydesc</code>)	268
1.01	General: Added range facility in		
	format key	96
	<code>\writeist</code> : Added spaces after		
	<code>\delimN</code> and <code>\delimR</code> in ist		
	file	142
1.03	<code>\makefirstuc</code> : changed 'pro-		
	ected@edef to 'def	241
1.04	General: Added <code>\glstextformat</code>		81
1.05	<code>\glossarysection</code> : added		
	<code>\mkboth</code> to <code>\glossarysection</code>	36
	<code>\gls@defglossaryentry</code> :		
	Changed the default value of		
	the sort key to just the value of		
	the name key	71
	<code>\glsmakefirstuc</code> : new	242
1.06	General: now requires <code>etoolbox</code>		240
	<code>\capitalisewords</code> : new	242
	<code>\xcapitalisewords</code> : new	243
1.07	<code>\gls@link</code> : fixed bug caused by		
	<code>\theglsentrycounter</code> set-		
	ting the page number too soon		94
	<code>\glsadd</code> : fixed bug caused by		
	<code>\theglsentrycounter</code> set-		
	ting the page number too soon	140
1.08	General: Added babel support	...	30
	<code>listgroup</code> : changed <code>listgroup</code>		
	style to use <code>\glsgetgrouptitle</code>	249
	<code>altlistgroup</code> : changed <code>al-</code>		
	<code>tlistgroup</code> style to use		
	<code>\glsgetgrouptitle</code>	250
	<code>\makefirstuc</code> : made robust	...	240
1.09	<code>\mfu@nocaplist</code> : new	243
	<code>\capitalisewords</code> : added check		
	for words that shouldn't be		
	capitalised	242
	<code>\gMFUnocap</code> : new	243
	<code>\mfu@checkword</code> : new	242
	<code>\MFUclear</code> : new	243
1.1	<code>\glossarysection</code> : numbered		
	sections and auto label added		37
	<code>\gls@tmpb</code> : changed <code>\toksdef</code>		
	to <code>\newtoks</code>	99
	<code>\gls@toc</code> : numberline added	..	38
	<code>\p@glossarysection</code> : num-		
	bered sections and auto label		
	added	37
	General: Added support for trans-		
	lator package	31
	<code>amsgen</code> now loaded (<code>\new@ifnextchar</code>		
	needed)	4
	<code>translate</code> : <code>translate</code> option		
	added	21
	<code>\setglossarysection</code> : new	...	37
	<code>numberedsection</code> : numbered-		
	section package option added	..	6
	<code>numberline</code> : <code>numberline</code> option		
	added	5

1.12	Removed restriction on only using \newglossaryentry in the preamble	75
\@GLSpl: now uses \glsentrydescplural and \glsentrysymbolplural instead of \glsentrydesc and \glsentrysymbol	110	\newacronym: Removed restriction on only using \newacronym in the preamble 193
\@GLspl@: now uses \glsentrydescplural and \glsentrysymbolplural instead of \glsentrydesc and \glsentrysymbol	109	1.14 \gls@hypergroup: new 245
\@GLspl@: now uses \glsentrydescplural and \glsentrysymbolplural instead of \glsentrydesc and \glsentrysymbol	109	General: added nonumberlist key to \printglossary 179
General: added check for \hypertarget separate to \hyperlink (memoir defines \hyperlink but not \hypertarget)	105	added numberedsection key to \printglossary 177
descriptionplural: new	58	\firstacronymfont: new 197
\gls@defglossaryentry: Changed default first plural to be first key with s appended (was text key with s appended)	71	\glsautoprefix: new 6
descriptionplural support added	70	\glsnavhyperlink: changed 'edef to 'protected@edef 244
symbolplural support added	70	\glsnavhypertarget: added write to aux file 244
\Glsentrydescplural: New	134	\glsnavigation: changed to only use labels for groups that are present 245
\Glsentrydescplural: New	133	1.15 \@gls@link: added \glslabel 94
\Glsentrysymbolplural: New	135	General: Added \glssettoctitle 31
\Glsentrysymbolplural: New	134	\gls@defglossaryentry: check for \@glo@first in description 74
\SetDescriptionFootnoteAcronymStyle: Added \protect before \footnote and \glslink	214	check for \@glo@text in symbol 75
\SetFootnoteAcronymStyle: Added \protect before \footnote and \glslink	220	\gls@hypergroup: new 245
symbolplural: new	59	\glsnavhypertarget: added check if rerun required 244
1.13	General: Add Polish support	349, 352
fixed bug that ignored 3rd parameter	112–120	\glssettoctitle: new 29
\ACRfullpl: new	196	\printglossary: changed the way the TOC title is set 164
\Acrfullpl: new	196	1.16 \@GLS@: Test glossary type is \acronymtype in addition to checking if footnote option has been used 108
\acrfullpl: new	195	\@GLSpl: Test glossary type is \acronymtype in addition to checking if footnote option has been used 110
\acrpluralsuffix: New	193	\@Gls@: Test glossary type is \acronymtype in addition to checking if footnote option has been used 107
\gls@defglossaryentry: Changed default first value	70	
Changed default firstplural value	71	

\@Glspl@: Test glossary type is \acronymtype in addition to checking if footnote option has been used	110	\glshyperlink:new	139
\@gls@: Test glossary type is \acronymtype in addition to checking if footnote option has been used	106	\glshypernumber: modified to allow material to be attached to location	190
\@glsdisp: Test glossary type is \acronymtype in addition to checking if footnote option has been used	111	\glsnavhyperlink: replaced 'hy- perlink to '@glslink	244
\@glspl@: Test glossary type is \acronymtype in addition to checking if footnote option has been used	109	\glsnavhypertarget: replaced 'hypertarget to '@glstarget .	244
\@glstarget: raised the hyper- target so the target text doesn't scroll off the top of the page	105	\glssee:new	162
\gls@defglossaryentry: Changed def to let	70	\glsseeformat:new	162
1.17		\glsSetSuffixF:new	34
\@@do@wrglossary:new	159	\glsSetSuffixFF:new	34
\@do@seeglossary:new	161	\ifglsexindy:new	23
\@glo@storeentry:new	76	\istfilename: added xindy sup- port	33
\@glossary: changed defini- tion to use \index instead of \@index	157	\newglossarystyle: made \newglossarystyle long .	189
\@glsdefaultplural:new	62	\nopostdesc:new	32
\@glsdefaultsort:new	62	nonumberlist:new	60
\@glshypernumber:new	190	\printglossary: added check to determine if \printglossary is already defined	164
\@glsnoname:new	61	added print language to aux file	164
\@glsnonextpages:new	180	order: order package option added	23
\@wrglossary: modified to allow for xindy support	157	\writeist: added xindy support	142
General: added Brazilian dictio- nary	353	1.18	
Added Brazilian support	349	\@gls@loadlist:new	8
added xindy support	23	\@gls@loadlong:new	8
parent:new	60	\@gls@loadsuper:new	8
see:new	60	\@gls@loadtree:new	8
\gls@defglossaryentry: added nonumberlist key	71	\gls@defglossaryentry: Changed default value of sort to \@glsdefaultsort	71
added parent key	71	moved sort sanitization to \newglossaryentry	74
added see key	71	\glstarget:new	183
Stored main part of entry format when entry is defined	75	\oldacronym:new	192
\gls@suffixF:new	34	nolist:new	8
\gls@suffixFF:new	34	nolong:new	8
		sort: moved sanitization to \newglossaryentry	58
		nostyles:new	8
		nosuper:new	8
		notree:new	8
		1.19	
		\glsclearpage:new	38
		\glsdisp:new	111

		2.03	
\SetDescriptionAcronymStyle:			\@GLS@: Added check for hyper-
changed \acronymfont to			first 108
use \textsmaller instead of			\@GLSpl: Added check for hyper-
\smaller 218			first 110
\SetDescriptionFootnoteAcronymStyle:			\@GLs@: Added check for hyper-
changed \acronymfont to			first 107
use \textsmaller instead of			\@GLspl@: Added check for hyper-
\smaller 214			first 110
\SetFootnoteAcronymStyle:			\@gls@: Added check for hyper-
changed \acronymfont to			first 106
use \textsmaller instead of			\@gls@@link: new 93
\smaller 221			\@gls@link: added \leavevmode
\SetSmallAcronymStyle:		 94
changed \acronymfont to			Moved entry existence check to
use \textsmaller instead of			avoid duplicate code 94
\smaller 224			\@glsdisp: Added check for hyper-
1.2	General: fixed bug in ngerman		hyperfirst 111
	captions 346		\@glspl@: Added check for hyper-
2.01			first 109
	\@gls@link: moved \@do@wrglossary		\@gls@: Added check
	before term is displayed to pre-		to see if it's already defined .. 36
	vent unwanted whatsit 95		hyperfirst: new 22
	\foralllglossaries: replaced	2.04	
	\ifthenelse with \ifx 48		\@GLS@: Changed test to check if
	\forallgsentries: replaced		glossary type has been identi-
	\ifthenelse with \ifx 48		fied as a list of acronyms ... 108
	\glsdefmain: new 12		\@GLSpl: Changed test to check if
	\glsdescwidth: changed		glossary type has been identi-
	\linewidth to \hsize . 252,267		fied as a list of acronyms ... 110
	\glslistdottedwidth: changed		\@GLs@: Changed test to check if
	\linewidth to \hsize 251		glossary type has been identi-
	\glspagelistwidth: changed		fied as a list of acronyms ... 107
	\linewidth to \hsize . 252,267		\@GLspl@: Changed test to check
	nomain: added nomain package		if glossary type has been iden-
	option 13		tified as a list of acronyms .. 110
	\writeist: removed item_02 - no		\@glossaryentryfield: new .. 76
	such makeindex key 146		\@glossarysubentryfield:
2.02			new 76
	\@printglossary: suppressed		\@gls@: Changed test to check if
	warning globally rather than		glossary type has been identi-
	locally 166		fied as a list of acronyms ... 106
	General: Changed Brazil to Brazil-		\@glsacronymlists: new 14
	ian 353		\@glsdisp: Changed test to check
	false will prevent automatic		if glossary type has been iden-
	loading of translator package 28		tified as a list of acronyms .. 111
	\glossarysection: changed		\@glspl@: Changed test to check
	\@mkboth to \glossarymark 36		if glossary type has been iden-
	\gls@: New 36		tified as a list of acronyms ... 109

\@newglossaryentryposthook:	Removed spurious brace. Patch provided by Sergiu Dotenco	111
new	76	
\@newglossaryentryprehook:	\writeist: Added \string before opening and closing braces. Patch provided by Segiu Dotenco	147
new	76	
acronymlists:new	15	
\DeclareAcronymList:new ...	14	
\DefineAcronymSynonyms:new	209	2.06
\gls@defglossaryentry: added		
user1-6 keys	71	
\glsadd: fixed bug that ignored counter	140	
\Glsentryuseri:new	136	
\Glsentryuseri:new	135	2.07
\Glsentryuserii:new	136	
\Glsentryuserii:new	136	
\Glsentryuseriii:new	136	
\Glsentryuseriii:new	136	
\Glsentryuseriv:new	136	3.0
\Glsentryuseriv:new	136	
\Glsentryuserv:new	136	
\Glsentryuserv:new	136	
\Glsentryuservi:new	137	
\Glsentryuservi:new	137	
\ns@newglossary: added check to determine if \gls@<type>@display and \gls@<type>@displayfirst have been defined.	55	
\SetAcronymLists:new	15	
\SetDefaultAcronymDisplayStyle:new	210	
\SetDefaultAcronymStyle:new	211	
\SetDescriptionAcronymDisplayStyle:new	216	
\SetDescriptionDUAAcronymDisplayStyle:new	215	
\SetDescriptionFootnoteAcronymDisplayStyle:new	212	
\SetDUADisplayStyle:new ..	224	
\SetFootnoteAcronymDisplayStyle:new	219	
\SetSmallAcronymDisplayStyle:new	221	
2.05		
\@glsdisp: Added closing brace. Patch provided by Sergiu Dotenco	111	
\altnewglossary:new	56	
\CustomAcronymFields:new ..	226	
\CustomNewAcronymDef:new ..	227	
\SetCustomDisplayStyle:new	226	
\SetCustomStyle:new	227	
General: glsadd format key stored in \@glsnumberformat (was mistakenly stored in \@glo@format)	140	
\@@do@wrglossary: added check for hyper location prefix ...	159	
modified to use new format ..	159	
\@@glossarysec: replaced \@ifundefined with \ifcsundef	5	
\@do@seeglossary: Sanitize and escape cross-referencing information	161	
\@gls@counterwithin:new	9	
\@gls@ifinlist:new	39	
\@gls@link: added \@gls@saveentrycounter	95	
added \@gls@setsort	95	
\@gls@saveentrycounter:new	95	
\@gls@setsort@def:new ...	11	
\@gls@setsort@standard:new	10	
\@gls@setsort@use:new ...	11	
\@gls@xdy@locationlist:new	42	
\@gls@link: replaced \@ifundefined with \ifcsundef	104	
\@glsnextpages:new	180	
\@makeglossary: Added check for savewrites	148	
\@print@glossary: replaced \@ifundefined with \ifcsundef	166	
\@printglossary: added \currentglossary	165	
added \@glsnextpages	165	

make toctitle default to title ..	165	\GlsAddXdyCounters:new	39
\@set@glo@numformat: added		\glentrycounterlabel:new	182
4th argument	97	\glentryitem:new	182
\@wrglossary: modified to take		\glentrylong:new	137
into account savewrites	157	\glentrylong:new	137
\@xdyattributelist:new	39	\glentrylongpl:new	137
General: added prefix to hyperlink		\glentrylongpl:new	137
.....	191	\glentryshort:new	137
etoolbox now loaded	4	\glentryshort:new	137
replaced \@ifundefined with		\glentryshortpl:new	137
\ifcsundef	28, 91, 177	\glentryshortpl:new	137
\acrfootnote:new	212	\glsggetgrouptitle: re-	
\ACRfull: added starred version	195	placed \@ifundefined with	
\Acrfull: added starred version	195	\ifcsundef	187
\acrfull: added starred version	194	\glsglossarymark: replaced	
\ACRfullpl: added starred ver-		\@ifundefined with	
sion	196	\ifcsundef	36
\Acrfullpl: added starred ver-		\glshyperlink: changed de-	
sion	196	fault from \glentryname to	
\acrfullpl: added starred ver-		\glentrytext	139
sion	195	\glshypernumber: replaced	
\acrlinkfootnote:new	212	\@ifundefined with	
\acrnolinkfootnote:new ...	212	\ifcsundef	190
\addglossarytocaptions: re-		\glnumberformat: replaced	
placed \@ifundefined with		\@ifundefined with	
\ifcsundef	30	\ifcsundef	34
savewrites:new	25	\glhrefentry:new	182
see: added \@glo@seeautonumberlist		\glresetsubentrycounter:	
.....	60	new	181
seeautonumberlist:new	7	\glseeitem: hyperlink uses	
\glossarysection: replaced		\glseeitemformat instead	
\@ifundefined with		of \glentryname	163
\ifcsundef	36	\glseeitemformat:new	163
\glossarystyle: replaced		\glssortnumberfmt:new	10
\@ifundefined with		\glssstepentry:new	181
\ifcsundef	188	\glssstepsubentry:new	181
\glsc@codepage: replaced		\glssubentrycounterlabel:	
\@ifundefined with		new	182
\ifcsundef	24	\glssubentryitem:new	182
\glsc@defglossaryentry: added		theglossary: replaced \@ifundefined	
\@glsc@defsort	74	with \ifcsundef	183
added short and long keys	71	short:new	61
replaced \@ifundefined with		shortplural:new	61
\ifcsundef	71	\ifglossaryexists: re-	
\glsc@doclearpage: replaced		placed \@ifundefined with	
\@ifundefined with		\ifcsundef	49
\ifcsundef	38	\ifglentryexists: re-	
\glscadd: added \@glsc@saveentrycounter		placed \@ifundefined with	
.....	140	\ifcsundef	49

\istfile: deprecated	156	\showglossaryin: new	233
glossaryentry: new	180	\showglossaryout: new	233
glossarysubentry: new	181	\showglossarytitle: new ...	233
\newglossaryentry: replaced		\showglosymbol: new	231
\DeclareRobustCommand		\showglosymbolplural: new .	231
with \newrobustcmd	64	\showglotext: new	229
\newglossarystyle: re-		\showglotype: new	229
placed \@ifundefined with		\showglouserii: new	230
\ifcsundef	189	\showglouseriii: new	230
\ns@newglossary: added		\showglouseriv: new	230
\@gls@defsortcount	55	\showglouservi: new	230
replaced \@ifundefined with		subentrycounter: new	9
\ifcsundef	55	\writeist: added xindy-only	
entrycounter: new	9	macro definitions to glossary	
entrycounterwithin: new	9	open tag	144
\oldacronym: replaced \@ifundefined		modified to support new for-	
with \ifcsundef	192	mat	142
compatible-2.07: compatible-			
2.07 option added	25		
long: new	61	3.01	
longplural: new	61	\@gls@writefiles: added check	
nonumberlist: now boolean ...	60	for empty glossaries	156
sort: new	10	General: made robust	107
counter: replaced \@ifundefined		\ACRfull: made robust	195
with \ifcsundef	59	\Acrfull: made robust	194
\printglossary: replaced		\acrfull: made robust	194
\@ifundefined with		\acrfullformat: removed	
\ifcsundef	164	\acronymfont as it should al-	
\SetDescriptionFootnoteAcronymDisplayStyle: be set in the second ar-		gument.	194
expanded options link op-		\ACRfullpl: made robust	196
tions	212	\Acrfullpl: made robust	196
\setentrycounter: added op-		\acrfullpl: made robust	195
tional argument	188	\ACRlong: made robust	130
\showacronymlists: new	232	\Acrlong: made robust	129
\showglocounter: new	229	\acrlong: made robust	128
\showglodesc: new	231	\ACRlongpl: made robust	132
\showglodescplural: new ...	231	\Acrlongpl: made robust	131
\showglofirst: new	229	\acrlongpl: made robust	130
\showglofirstpl: new	229	\ACRshort: made robust	126
\showgloflag: new	232	\Acrshort: made robust	126
\showgloindex: new	232	\acrshort: made robust	125
\showglolevel: new	228	\ACRshortpl: made robust	128
\showgloname: new	231	\Acrshortpl: made robust	127
\showgloparent: new	228	\acrshortpl: made robust	127
\showgloplural: new	229	\Gls: made robust	107
\showglosort: new	231	\glsadd: made robust	140
\showglossaries: new	233	\glsaddall: made robust	140
\showglossarycounter: new .	233	\GLSdesc: made robust	117
\showglossaryentries: new .	233		

\Glsdesc: made robust	116	3.02	
\glsdesc: made robust	116		\@do@wrglossary: changed
\GLSdescplural: made robust .	118		\@glslocref to \theglsentrycounter
\Glsdescplural: made robust .	117	 160
\glsdescplural: made robust .	117		\@do@wrglossary: changed
\glsfirst: made robust	112		\@do@wr@glossary to test for
\GLSfirstplural: made robust	115		indexonlyfirst option; put old
\Glsfirstplural: made robust	115		\@do@wr@glossary code into
\glsfirstplural: made robust	114		\@do@wrglossary
\glslink: made robust	93		158
\GLSname: made robust	116		\@gls@missingnumberlist:
\Glsname: made robust	115		new
\glsname: made robust	115		62
\GLSpl: made robust	110		\@glswritefiles: added check
\Glspl: made robust	109		for existence of token in case
\glspl: made robust	108		\makeglossaries has been
\GLSplural: made robust	114		omitted
\GLSsymbol: made robust	118		156
\Glsymbol: made robust	118		\@printglossary: add a way to
\GLSsymbolplural: made robust			fetch current entry label ...
.....	119		166
\Glsymbolplural: made robust			\@wrglossary: added check for
.....	119		glossary file defined
\glsymbolplural: made robust			158
.....	119		General: added check for polyglos-
\Glstext: made robust	112		sia
\glstext: made robust	112		28
\GLSuseri: made robust	120		reversed order of package check
\Glsuseri: made robust	120		32
\glsuseri: made robust	120		savenumberlist: new
\GLSuserii: made robust	121		7
\Glsuserii: made robust	121		ucmark: new
\glsuserii: made robust	121		9
\GLSuseriii: made robust	122		\@gls@defglossaryentry: added
\Glsuseriii: made robust	122		numberlist element
\glsuseriii: made robust	121		74
\GLSuseriv: made robust	123		\@gls@save@numberlist: new ..
\Glsuseriv: made robust	123		163
\glsuseriv: made robust	122		\@glsdisplaynumberlist: new
\GLSuseriv: made robust	124		138
\Glsuseriv: made robust	123		\@glsentrycounter: set default
\GLSuseriv: made robust	124		value
\glsuseriv: made robust	122		95
\GLSuseriv: made robust	124		\@glsentryfull: fixed bug (re-
\Glsuseriv: made robust	123		placed \@glsentryshortpl
\GLSuseriv: made robust	124		with \@glsentryshort)
\glsuseriv: made robust	122		138
\GLSuseriv: made robust	124		\@glsentryfullpl: fixed bug (re-
\Glsuseriv: made robust	123		placed \@glsentryshort with
\GLSuseriv: made robust	124		\@glsentryshortpl)
\glsuseriv: made robust	122		138
\GLSuseriv: made robust	124		\@glsentrynumberlist: new ..
\Glsuseriv: made robust	123		138
\GLSuseriv: made robust	124		\@glsmoveentry: new
\glsuseriv: made robust	122		76
\GLSuseriv: made robust	124		\@glsnumlistlastsep: new ...
\Glsuseriv: made robust	123		139
\GLSuseriv: made robust	124		\@glsnumlistsep: new
\glsuseriv: made robust	122		139
\GLSuseriv: made robust	124		\@glsresetsubentrycounter:
\Glsuseriv: made robust	123		new
\GLSuseriv: made robust	124		181
\glsuseriv: made robust	122		\@ifglshaschildren: new
\GLSuseriv: made robust	124		50
\Glsuseriv: made robust	123		\@ifglshasparent: new
\GLSuseriv: made robust	124		51
\glsuseriv: made robust	122		\@makeglossaries: added list
\GLSuseriv: made robust	124		parser
\Glsuseriv: made robust	123		151
\GLSuseriv: made robust	124		indexonlyfirst: new
\glsuseriv: made robust	122		22
\GLSuseriv: made robust	124		\@renewglossarystyle: new ..
\Glsuseriv: made robust	123		189

\showglossaryentries: fixed misspelt command	233	super3col: added check for glsnogroupskip	270
\SmallNewAcronymDef: fixed broken short and long plural	222	super4col: added check for glsnogroupskip	271
3.03		superragged: added check for glsnogroupskip	275
\@gls@sanitizesort:new	18	superragged3col: added check for glsnogroupskip	277
\@gls@setupsort@standard: used \@gls@sanitizesort .	10	3.04	
\@printglossary: allow title to override default toctitle	165	\@do@wrglossary: changed \theglsentrycounter back to \@glslocref	160
General: allow title to set toctitle	177	\@do@wrglossary: modified to compensate for possible incorrect page number	159
\glsinlinedescformat:new .	248	\@gls@escbsdq: unsanitize \gls@numberpage, \gls@alphpage, \gls@Alphpage and \gls@romanpage	98
\glsinlineemptydescformat:new	248	\@print@glossary: Moved aux write to end of document to prevent unwanted whatsit occurring here.	166
\glsinlinenameformat:new .	248	General: Added check for doc package	4
\glsinlinepostchild:new ..	248	added datatool-base as a required package	4
\glsinlinesubdescformat:new	248	added local key	92
\glsinlinesubnameformat:new	248	\gls@Alphpage:new	158
\glspostinline: replaced “.” with \glspostdescription	248	\gls@alphpage:new	158
atlongragged4col: added check for glsnogroupskip ..	262	\gls@disablepagerefexpansion:new	158
altsuperragged4col: added check for glsnogroupskip ..	278	\gls@numberpage:new	158
alttree: added check for glsnogroupskip	286	\gls@protected@pagefmts:new	158
index: added check for glsnogroupskip	281	\gls@romanpage:new	159
nogroupskip:new	9	\glsdefmain: added check for doc package	12
long: added check for glsnogroupskip	252	\glsorg@endtheglossary:new .	5
long3col: added check for glsnogroupskip	254	\glsorg@glossary:new	4
long4col: added check for glsnogroupskip	255	\glsorg@theglossary:new	5
longragged: added check for glsnogroupskip	259	\glsorg@wrglossary:new	4
longragged3col: added check for glsnogroupskip	260	altlist: replaced \newline with paragraph break	250
nopostdot:new	9	\PrintChanges:new	5
tree: added check for glsnogroupskip	282	3.05	
treenoname: added check for glsnogroupskip	284	\@do@wrglossary: add Roman case. Fixed bugs in the else statements	159
super: added check for glsnogroupskip	268		

\@gls@link: added check for “no-hypertypes”	94	General: added nogroupskip key to \printglossary	178
\@gls@nohyperlist: new	16	removed definition of \@glossaryentryfield ..	327
mcolalttree: replaced ‘2’ with \glsmcols	266	removed definition of \@glossarysubentryfield	327
mcolindex: replaced ‘2’ with \glsmcols	264	\compatibleglossentry: new	183
mcoltree: replaced ‘2’ with \glsmcols	264	\compatiblesubglossentry: new	185
mcoltreenoname: replaced ‘2’ with \glsmcols	265	\glossaryentryfield: deprecated	185
\gls@protected@pagefmts: added Roman to list	158	\Glossentrydesc: new	184
\gls@Romanpage: new	159	\glossentrydesc: new	184
\GlsDeclareNoHyperList: new	16	\Glossentryname: new	184
\glsgetgrouplabel: fixed bug (typo in \equal)	188	\glossentryname: new	184
\nopostdesc: made robust	32	\Glossentrysymbol: new	184
nohypertypes: new	16	\glossentrysymbol: new	184
3.06		\gls@assign@desc@field: new	17
\@xdy@main@language: Changed back to using \language	24	\gls@assign@desc@plural@field: new	17
\findrootlanguage: Obsoleted	46	\gls@assign@field: new	64
3.07		\gls@ifnotmeasuring: new ...	78
\@gls@link: fixed bug that failed to find entry in list	94	\glsaddallunused: new	140
\glossarypreamble: modified to work with \setglossarypreamble	35	\glsexpandfields: new	64
\gls@doclearpage: added check for openright	38	\glsnoexpandfields: new	64
\glspostdescription: Added spacefactor code	8	\glssee: made robust	162
\GlsSetXdyCodePage: Added check for fontspec	47	\glsseeformat: made robust ..	162
\SetDescriptionAcronymDisplayStyle: now using \glsdoparenifnotempty	216	\glsseeitem: made robust	163
\setglossarypreamble: new ..	35	\glsseelist: made robust	162
3.08a		\ifglstdescsuppressed: new ..	51
\@gls@storeentry: no longer need to check for special characters in any of the fields other than sort	77	\ifglshasdesc: new	51
updated for \glossentry	77	\ifglshassymbol: new	51
\@glossaryentryfield: switched to \glossentry	76	list: updated list style to use \glossentry and \subglossentry	249
\@glossarysubentryfield: switched to \subglossentry	76	listdotted: updated listdotted style to use \glossentry and \subglossentry	251
		altlist: updated altlist style to use \glossentry and \subglossentry	250
		altlongragged4col: updated to use \glossentry and \subglossentry	261
		alttree: updated to use \glossentry and \subglossentry	285
		index: added paragraph break at end of environment	280

updated to use \glossentry and \subglossentry	280	\Glsentryuseriv: made robust .	136
inline: updated inline style to use \glossentry and \subglossentry	246	\Glsentryuseriv: made robust	137
long: updated to use \glossentry and \subglossentry	252	\glstextup: new	193
longragged: updated to use \glossentry and \subglossentry	258	\if@gl@docloaded: Add a fix for \RecordChanges	4
longragged3col: updated to use \glossentry and \subglossentry	260	\ifglshassymbol: changed test to check for \@gl@default@symbol	51
tree: updated to use \glossentry and \subglossentry	282	3.10a	
\setglossarystyle: new	188	\@gl@keymap: new	66
\setglossentrycompatibility: new	185	\@gl@provide@newglossary: new	53
superragged: updated to use \glossentry and \subglossentry	275	\@gl@writedef: new	65
3.09a		\@gl@defaultplural: Obsolete .	62
\@gl@assign@symbolplural@field: new	18	\@gl@nodesc: new	62
\@gl@default@value: new ...	59	\@print@glossary: Added providecommand code to aux file	167
\Glsentrydesc: made robust ..	133	\gl@assign@type@field: new	17
\Glsentrydescplural: made ro- bust	134	\gl@defglossaryentry: Changed to using \@gl@default@value	70, 71
\Glsentryfirst: made robust .	135	new	70
\Glsentryfirstplural: made robust	135	\glswritedefhook: new	69
\Glsentryfull: made robust ..	138	\makeglossaries: Added providecommand code to aux file	150
\Glsentryfullpl: made robust	138	\new@glossaryentry: new	65
\Glsentrylong: made robust ..	137	\ns@newglossary: added \@gl@provide@newglossary	55
\Glsentrylongpl: made robust	137	3.11a	
\Glsentryname: made robust ..	133	\@ACRlong: added \glslabel, \gl@ifplural, \glscapscase, \gl\$insert and \glscustomtext	326
\Glsentryplural: made robust	134	\@ACRshort: added \glslabel, \gl@ifplural, \glscapscase, \gl\$insert and \glscustomtext	325
\Glsentryshort: made robust .	137	\@Acrlong: added \glslabel, \gl@ifplural, \glscapscase, \gl\$insert and \glscustomtext	326
\Glsentryshortpl: made robust	137	\@Acrshort: added \glslabel, \gl@ifplural, \glscapscase, \gl\$insert and \glscustomtext	325
\Glsentrysymbol: made robust	134		
\Glsentrysymbolplural: made robust	135		
\Glsentrytext: made robust ..	134		
\Glsentryuseri: made robust .	136		
\Glsentryuserii: made robust	136		
\Glsentryuseriii: made robust	136		
\Glsentryuseriv: made robust	136		

<p>\@GLS@: add \glslabel, \glsifplural,\glscapscase, \glscustomtext and \glsinsert 108 change to using \glsentryfmt style commands 108 removed \MakeUppercase (now moved to \glsentryfmt) 108</p> <p>\@GLSpl: add \glslabel, \glsifplural,\glscapscase, \glscustomtext and \glsinsert 110 change to using \glsentryfmt style commands 110 removed \MakeUppercase as now dealt with in \glsentryfmt 110</p> <p>\@GLs@: add \glsifplural, \glscapscase,\glscustomtext and \glsinsert 107 change to using \glsentryfmt style commands 107 removed \makefirstuc (now dealt with in \glsentryfmt) 107</p> <p>\@GLspl@: add \glsifplural, \glscapscase,\glscustomtext and \glsinsert 109 change to using \glsentryfmt style commands 109 removed \makefirstuc (now dealt with in \glsentryfmt) 110</p> <p>\@acrlong: added \glslabel, \glsifplural,\glscapscase, \glsinsert and \glscustomtext 326</p> <p>\@acrshort: added \glslabel, \glsifplural,\glscapscase, \glsinsert and \glscustomtext 325</p> <p>\@gls@: add \glslabel, \glsifplural,\glscapscase, \glscustomtext and \glsinsert 106 change to using \glsentryfmt style commands 106</p> <p>\@gls@noexpand@fields: Fixed bug expand replaced with noexpand 63</p>	<p>\@glsdisp: add \glslabel, \glsifplural,\glscapscase, \glscustomtext and \glsinsert 111 change to using \glsentryfmt style commands 111</p> <p>\@glspl@: add \glslabel, \glsifplural,\glscapscase, \glscustomtext and \glsinsert 109 change to using \glsentryfmt style commands 109</p> <p>General: added \glslabel, \glsifplural,\glscapscase, \glsinsert and \glscustomtext 125–132 changed to just use \Glsentrydescplural 117 changed to just use \glsentrydescplural 117, 118 changed to just use \Glsentrydesc 117 changed to just use \glsentrydesc 116, 117 changed to just use \Glsentryfirstplural 115 changed to just use \glsentryfirstplural 114, 115 changed to just use \Glsentryfirst 113 changed to just use \glsentryfirst 113 changed to just use \Glsentryname 116 changed to just use \glsentryname 115, 116 changed to just use \Glsentryplural 114 changed to just use \glsentryplural 114 changed to just use \Glsentrysymbolplural 119 changed to just use \glsentrysymbolplural 119, 120 changed to just use \Glsentrysymbol 118 changed to just use \glsentrysymbol 118, 119</p>
--	---

Changed to just use	\glsglossarymark: replaced
\Glsentrytext 112	\MakeUppercase with
changed to just use \glentrytext	\mfirstucMakeUppercase . 36
..... 112	\glsnavigation: switched to us-
changed to just use \Glsentryuseriii	ing \@gls@getgrouptitle 245
..... 122	\ifglshasdesc: replaced
changed to just use \glentryuseriii	\ifdefempty with \ifcsemtyp
..... 122 51
changed to just use \Glsentryuserii	\ifglshaslong: new 51
..... 121	\ifglshasshort: new 52
changed to just use \glentryuserii	\ifglshassymbol: replaced
..... 121	\ifdefempty with \ifcsemtyp
changed to just use \Glsentryuseriv 51
..... 123	\ifglused: replaced \ifthenelse
changed to just use \glentryuseriv	with \ifbool 49
..... 123	\longnewglossaryentry: new . 69
changed to just use \Glsentryuseri	\ns@newglossary: replaced
..... 120	\glssdisplay and \glssdisplayfirst
changed to just use \glentryuseri	with \glssentryfmt 55
..... 120	compatible-3.07: cnew 25
changed to just use \Glsentryuservi	\SetCustomDisplayStyle: up-
..... 125	dated to use \defglssentryfmt
changed to just use \glentryuservi 226
..... 124, 125	\SetDefaultAcronymDisplayStyle:
changed to just use \Glsentryuserv	changed to use \defglssentryfmt
..... 124 210
changed to just use \glssentryuserv	\SetDescriptionAcronymDisplayStyle:
..... 123, 124	updated to use \defglssentryfmt
Now requires textcase 3 216
acronymlists: replaced	\SetDescriptionDUAAcronymDisplayStyle:
\@addtoacronymlists with	updated to use \defglssentryfmt
\DeclareAcronymList 15 215
\defglssdisplay: obsoleted 90	\SetDescriptionFootnoteAcronymDisplayStyle:
\defglssdisplayfirst: obso-	updated to use \defglssentryfmt
leted 90 212
\defglssentryfmt: new 54	\SetDUADisplayStyle: updated
\forglssentries: replaced \ifx	to use \defglssentryfmt .. 224
with \ifdefempty 48	\SetFootnoteAcronymDisplayStyle:
\gls@assign@desc: new 69	updated to use \defglssentryfmt
\gls@defglossaryentry: Fixed 219
default counter if none sup-	\SetSmallAcronymDisplayStyle:
plied 74	updated to use \defglssentryfmt
\gls@doentryfmt: new 54 221
\glssdisplay: obsoleted 90	\setupglossaries: new 27
\glssdisplayfirst: obsoleted .. 90	\showglolong: new 232
\glssgenentryfmt: new 85	\showgloshort: new 232
\glssgetgrouptitle: Added	numbers: new 26
check in case non-Latin alpha-	symbols: new 25
bet in use 187	

3.12a	\gls@defglossaryentry: added	\glsentryfullpl: changed to use \acrfullformat 138
	\glslabel 70	\gls glossarymark: renamed
	\glsaddkey: new 67	\glossarymark to \gls glossarymark to avoid conflict with memoir 36
3.13a	\@gls@assign@symbol@field:	\glsprestandardsort: new ... 10
	changed to use \glssetnoexpandfield	\glssetexpandfield: new 17
 18	\glssetnoexpandfield: new .. 17
	\@gls@assign@symbolplural@field:	altsuper4colheader: switched
	changed to use \glssetnoexpandfield	to \tabularnewline 273
 18	altsuper4colheaderborder:
	\@gls@link: removed \relax .. 95	switched to \tabularnewline
	\@gls@notranslatorhook: new 21 273
	\@gls@setupsort@standard:	long: switched to \tabularnewline
	moved \@gls@santizesort 252
	to \glsprestandardsort .. 10	long3col: switched to \tabularnewline
	General: added cs@gls@notranslatorhook 254
	to else clause 32	long3colheader: switched to
	ucmark: added check for memoir . 9	\tabularnewline 254
	see: added \gls@checkseeallowed	long3colheaderborder: switched
 60	to \tabularnewline 255
	\glossarysection: changed	long4col: switched to \tabularnewline
	\glossarymark to \gls glossarymark 255
 36	long4colheader: switched to
	\glossarystyle: fixed bug	\tabularnewline 256
	caused by using \ifdef in-	longheader: switched to
	stead of \ifcsdef 188	\tabularnewline 253
	\gls@assign@desc@field:	longheaderborder: switched to
	changed to use \glssetnoexpandfield	\tabularnewline 253
 17	\SetFootnoteAcronymDisplayStyle:
	\gls@assign@descplural@field:	fixed missing argument bug 219
	changed to use \glssetnoexpandfieldsuper:	switched to \tabularnewline
 17 268
	\gls@assign@name@field:	super3col: switched to
	changed to use \glssetnoexpandfield	\tabularnewline 269
 17	super3colheader: switched to
	\gls@assign@type@field:	\tabularnewline 270
	changed to use \glssetexpandfield	super4col: switched to
 17	\tabularnewline 271
	\gls@checkseeallowed: new .. 60	super4colheader: switched to
	\glsaddallunused: set default to	\tabularnewline 271
	\@glo@types 140	super4colheaderborder:
	\Glsentryfull: changed to use	switched to \tabularnewline
	\acrfullformat 138 272
	\glsentryfull: changed to use	superheader: switched to
	\acrfullformat 138	\tabularnewline 268
	\Glsentryfullpl: changed to	superheaderborder: switched to
	use \acrfullformat 138	\tabularnewline 269

3.14a		\genacrfullformat:new 89
\@glswritefiles:	renamed	\GenericAcronymFields:new 198
\glswritefiles to \@glswritefiles		\Genplacrfullformat:new . . . 90
and used “savewrites” option		\genplacrfullformat:new . . . 89
to set \@glswritefiles	156	\Glsentryfull: bug fix: added
General:new	234	missing\acronymfont 138
acronyms:new	14	\Glsentryfull: bug fix: added
\gls@defglossaryentry: added		missing\acronymfont 138
check for existence of default		\Glsentryfullpl: bug fix: added
glossary	71	missing\acronymfont 138
set the default for firstplural to		\Glsentryfullpl: bug fix: added
be the value of plural	73	missing\acronymfont 138
xindygloss:new	24	\glsgenacfmt:new 87
\longprovideglossaryentry:		\GlsUseAcrEntryDisplayStyle:
new	70	new 200
compatible-2.07: added check		\GlsUseAcrStyleDefs:new . . 200
for 2.07 before setting 3.07		short-long:new 200
compatibility	25	short-long-desc:new 203
notranslate:new	21	xindynoglsnumbers:new 24
\provideglossaryentry:new	64	sm-short-long:new 202
4.0		sm-short-long-desc:new . . . 203
\gls@defglossaryentry: added		\makeglossaries: made pream-
check for first key	73	ble only 151
4.01		index:new 26
General: fixed non-value options		\newacronymstyle:new 199
so that they can be passed to		long-sc-short:new 201
document class	7	long-sc-short-desc:new . . . 202
\CustomAcronymFields:	in-	long-short:new 200
serted missing comma	227	long-short-desc:new 202
4.02		long-sm-short:new 201
\@acrfull: now using \acrfullfmt		long-sm-short-desc:new . . . 203
.	194	footnote:new 206
\@gls@indexdef:new	26	footnote-desc:new 208
\@gls@numbersdef:new	26	footnote-sc:new 207
\@gls@symbolsdef:new	26	footnote-sc-desc:new 208
General: Removed \acronymfont		footnote-sm:new 207
.	129–132	footnote-sm-desc:new 208
\ACRfullfmt:new	195	\setacronymstyle:new 199
\Acrfullfmt:new	195	\SetDescriptionAcronymDisplayStyle:
\acrfullfmt:new	194	Moved check for empty cus-
\ACRfullplfmt:new	196	tom text to prevent unwanted
\Acrfullplfmt:new	196	parenthetical material 216
\acrfullplfmt:new	196	\SetDescriptionFootnoteAcronymDisplayStyle:
\acronymentry:new	198	Moved check for empty cus-
sanitize: fixed bug that caused		tom text to prevent unwanted
an error here	21	parenthetical material 212
sc-short-long:new	202	\SetFootnoteAcronymDisplayStyle:
sc-short-long-desc:new	203	Moved check for empty cus-
\Genacrfullformat:new	89	

tom text to prevent unwanted parenthetical material	219	\@glsdisp: removed \glslabel (defined in \@gls@link) . . .	111
\SetGenericNewAcronym: new	197	\@glspl@: removed \glslabel (defined in \@gls@link) . . .	109
\SetSmallAcronymDisplayStyle: Moved check for empty cus- tom text to prevent unwanted parenthetical material	221	\@printglossary: added \glsdetoklabel	166
dua: new	204	General: changed default to \@empty instead of \relax . .	25
dua-desc: new	206	removed \glslabel (defined in \@gls@link)	125
numberedsection: added		sc-short-long-desc: redefined to use accessibility informa- tion	331
nameref option	6	\compatibleglossentry: added \glsdetoklabel	307
4.03		\compatiblesubglossentry: added \glsdetoklabel . . .	308
\@do@wrglossary: added		\Genacrfullformat: redefined to use accessibility informa- tion	324
\glsdetoklabel	160	\genacrfullformat: redefined to use accessibility informa- tion	324
\@ACRlong: removed \glslabel (defined in \@gls@link) . . .	326	\Genplacrfullformat: rede- fined to use accessibility in- formation	324
\@ACRshort: removed \glslabel (defined in \@gls@link) . . .	325	\genplacrfullformat: rede- fined to use accessibility in- formation	324
\@Acrlong: removed \glslabel (defined in \@gls@link) . . .	326	\glossentryname: added \glsdetoklabel	184
\@Acrshort: removed \glslabel (defined in \@gls@link) . . .	325	\gls@defglossaryentry: added \glsdetoklabel	70
\@GLS@: removed \glslabel (de- fined in \@gls@link)	108	replaced #1 with \@glo@label	71
\@GLSpl: removed \glslabel (defined in \@gls@link) . . .	110	replaced \ifthenelse with \ifdefequal	72
\@Gls@: removed \glslabel (de- fined in \@gls@link)	107	\glsadd: added \glsdetoklabel	140
\@Gls@entry@field: new . . .	133	\glsaddkey: switched to using \@gls@field@link	68
\@Glspl@: removed \glslabel (defined in \@gls@link) . . .	109	\glsdetoklabel: new	49
\@acrlong: removed \glslabel (defined in \@gls@link) . . .	326	\glsdisplaynumberlist: added \glsdetoklabel	139
\@acrshort: removed \glslabel (defined in \@gls@link) . . .	325	\glsdoifexistsorwarn: new . .	50
\@gls@: removed \glslabel (de- fined in \@gls@link)	106	\glsentryaccess: switched to using \@gls@entry@field .	312
\@gls@access@display: new .	313	\glsentrydescaccess: switched to using \@gls@entry@field	312
\@gls@entry@field: new . . .	132		
\@gls@fetchfield: new	66		
\@gls@field@link: new	111		
\@gls@link: added \glsdetoklabel	94		
moved \@gls@link@opts and \@gls@link@label to \@gls@link	94		
\@gls@writedef: added \glsdetoklabel	65		

<code>\glsentrydescpluralaccess:</code> switched to using <code>\@gls@entry@field</code> 312	<code>\glsstepentry:added\glsdetoklabel</code> 181
<code>\glsentryfirstaccess:switched</code> to using <code>\@gls@entry@field</code> 312	<code>\glsstepsubentry: added</code> <code>\glsdetoklabel</code> 181
<code>\glsentryfirstplural: added</code> <code>\glsdetoklabel</code> 135	<code>\glsunset:added\glsdetoklabel</code> 79
<code>\glsentrylongaccess: switched</code> to using <code>\@gls@entry@field</code> 313	short-long: commented spuri- ous EOL 201
<code>\glsentrylongpluralaccess:</code> switched to using <code>\@gls@entry@field</code> 313	redefined to use accessibility in- formation 329
<code>\glsentrypluralaccess:</code> switched to using <code>\@gls@entry@field</code> 312	short-long-desc: redefined to use accessibility information 331
<code>\glsentryshortaccess:switched</code> to using <code>\@gls@entry@field</code> 313	<code>\ifglsdescsuppressed: added</code> <code>\glsdetoklabel</code> 51
<code>\glsentryshortpluralaccess:</code> switched to using <code>\@gls@entry@field</code> 313	fixed typo 51
<code>\glsentrysymbolaccess:</code> switched to using <code>\@gls@entry@field</code> 312	<code>\ifglsentryexists: added</code> <code>\glsdetoklabel</code> 49
<code>\glsentrysymbolpluralaccess:</code> switched to using <code>\@gls@entry@field</code> 312	<code>\ifglsdescchildren: added</code> <code>\glsdetoklabel</code> 50
<code>\glsentrytextaccess: switched</code> to using <code>\@gls@entry@field</code> 312	<code>\ifglsdesc:added\glsdetoklabel</code> 51
<code>\glsacacmt: redefined to use</code> accessibility information ... 322	<code>\ifglsdescfield:new</code> 52
<code>\glsacentryfmt: redefined to</code> use accessibility information 319	<code>\ifglsdesc:added\glsdetoklabel</code> 51
<code>\glsachyperlink:added\glsdetoklabel</code> 139	<code>\ifglsdescparent: added</code> <code>\glsdetoklabel</code> 51
<code>\glslocalreset: added</code> <code>\glsdetoklabel</code> 78	<code>\ifglsdescshort: added</code> <code>\glsdetoklabel</code> 52
<code>\glslocalunset: added</code> <code>\glsdetoklabel</code> 79	<code>\ifglsdescsymbol: added</code> <code>\glsdetoklabel</code> 51
<code>\glsmoveentry:added\glsdetoklabel</code> 76	replaced <code>\ifcsemt</code> with <code>\ifdefempty</code> and replaced <code>\ifx</code> with <code>\ifdefequal</code> 51
replaced <code>\ifthenelse</code> with <code>\ifdefequal</code> 76	<code>\ifglsused:added\glsdetoklabel</code> 49
<code>\glsrefentry:added\glsdetoklabel</code> 182	sm-short-long-desc: redefined to use accessibility informa- tion 331
<code>\glsreset:added\glsdetoklabel</code> 78	long-sc-short-desc: redefined to use accessibility informa- tion 330
<code>\glsseelist:added\expandafter</code> commands 163	long-short: redefined to use ac- cessibility information 328
	long-short-desc: redefined to use accessibility information 330
	long-sm-short-desc: redefined to use accessibility informa- tion 330

footnote: redefined to use accessibility information	334	\showgloplural: added	
footnote-desc: redefined to use accessibility information . . .	336	\glsdetoklabel	229
footnote-sc: redefined to use accessibility information	336	\showglopluralaccess: added	
footnote-sc-desc: redefined to use accessibility information	337	\glsdetoklabel	343
footnote-sm: redefined to use accessibility information	336	\showgloshort: added \glsdetoklabel	
footnote-sm-desc: redefined to use accessibility information	337	232
\renewacronymstyle: new . . .	199	\showgloshortaccess: added	
\showglocounter: added		\glsdetoklabel	343
\glsdetoklabel	229	\showgloshortpluralaccess:	
\showgloDESC: added \glsdetoklabel		added \glsdetoklabel . . .	343
.....	231	\showglosort: added \glsdetoklabel	
\showgloDESCaccess: added		231
\glsdetoklabel	343	\showglosymbol: added	
\showgloDESCplural: added		\glsdetoklabel	231
\glsdetoklabel	231	\showglosymbolaccess: added	
\showgloDESCpluralaccess:		\glsdetoklabel	343
added \glsdetoklabel . . .	343	\showglosymbolplural: added	
\showgloFIRST: added \glsdetoklabel		\glsdetoklabel	231
.....	229	\showglosymbolpluralaccess:	
\showgloFIRSTaccess: added		added \glsdetoklabel . . .	343
\glsdetoklabel	343	\showglotext: added \glsdetoklabel	
\showgloFIRSTpl: added		229
\glsdetoklabel	229	\showglotextaccess: added	
\showgloFIRSTpluralaccess:		\glsdetoklabel	343
added \glsdetoklabel . . .	343	\showglotype: added \glsdetoklabel	
\showgloFLAG: added \glsdetoklabel		229
.....	232	\showgloUSERI: added \glsdetoklabel	
\showgloINDEX: added \glsdetoklabel		230
.....	232	\showgloUSERII: added	
\showgloLEVEL: added \glsdetoklabel		\glsdetoklabel	230
.....	228	\showgloUSERIII: added	
\showgloLONG: added \glsdetoklabel		\glsdetoklabel	230
.....	232	\showgloUSERIV: added	
\showgloLONGaccess: added		\glsdetoklabel	230
\glsdetoklabel	344	\showgloUSERV: added \glsdetoklabel	
\showgloLONGpluralaccess:		230
added \glsdetoklabel . . .	344	\showgloUSERVI: added	
\showgloNAME: added \glsdetoklabel		\glsdetoklabel	230
.....	231	dua: fixed bug in \acrfullfmt .	205
\showgloNAMEaccess: added		fixed bug in \Acrfullplfmt .	205
\glsdetoklabel	342	fixed bug in \acrfullplfmt .	205
\showgloPARENT: added		redefined to use accessibility in-	
\glsdetoklabel	228	formation	331
		dua-desc: commented spurious	
		EOL	206
		redefined to use accessibility in-	
		formation	334

4.04	
\@gls@noidx@nosanitizesort:	\def as is may or may not be defined 307
new 18	\compatiblesubglossentry:
\@gls@noidx@sanitizesort:	changed \newcommand to
new 18	\def as is may or may not be defined 308
\@gls@nosanitizesort:new . 18	\defglsdisplayfirst: fixed unwanted space 91
\@gls@sanitizesort:new ... 18	\glo@grabfirst:new 173
\@glo@addchildren:new 168	\gls@defglossaryentry: replaced \ifx with \ifdefvoid 75
\@glo@do@sortentries:new . 169	\glsnoidxdisplayloc:new .. 176
\@glo@grabfirst:new 174	\glsnoidxdisplaylocclisthandler:
\@glo@sortedinsert:new ... 169	new 176
\@glo@sortentries:new 168	\glsnoidxloclist:new 175
\@glo@sorthandler@case:new 170	\glsnoidxlocclisthandler:
\@glo@sorthandler@letter:	new 176
new 170	\glsnoidxstripaccents:new . 19
\@glo@sorthandler@nocase:	alttree: moved hangindent and
new 170	parindent assignments outside level test 285
\@glo@sorthandler@word:new 170	\makeglossaries: Moved definition of \glswrite to
\@glo@sortmacro@case:new . 171	\makeglossaries 150
\@glo@sortmacro@def:new .. 172	\makenoidxglossaries:new . 152
\@glo@sortmacro@def@do:new 172	\printglossary: changed to use
\@glo@sortmacro@letter:new 171	new \@printglossary ... 164
\@glo@sortmacro@nocase:new 172	\printnoidxglossaries:new 164
\@glo@sortmacro@standard:	\printnoidxglossary:new .. 164
new 171	\showgloclist:new 232
\@glo@sortmacro@use:new .. 173	\warn@noprintglossary: Activate warning in \makeglossaries
\@glo@sortmacro@word:new . 170 164
\@gls@getcounterprefix:	\writeist: checked for definition
added warning if no prefix can	of \glswrite 142, 146
be formed 161	4.06
\@gls@getothergrouptitle:	\@GLS@: added \glsifhyper .. 108
new 187	\@GLSpl: added \glsifhyper . 110
\@gls@noidx@do:new 174	\@Gls@: added \glsifhyper .. 107
\@gls@noref@warn:new 155	\@Glspl@: added \glsifhyper 109
\@gls@reference:new 176	\@gls@: added \glsifhyper .. 106
\@gls@warnonglossdefined:	\@gls@numbersdef: added hook
new 16	to set toc title 26
\@gls@warnontheGLOSSdefined:	\@gls@symbolsdef: added hook
new 16	to set toc title 26
\@no@makeglossaries:new .. 155	\@glsdisp: added \glsifhyper 111
\@print@glossary:new 166	\@glspl@: added \glsifhyper 109
\@print@noidx@glossary:new 173	General: added \glsifhyper ..
\@printgloss@setsort:new . 164 126–132
\@printglossary:new 165	
General: added sort key to print-	
gloss group 179	
\compatibleglossentry:	
changed \newcommand to	

acronym: added hook to set toc title	13	\@gls@forbidtexext: new	54
acronyms: added hook to set toc title	14	\@gls@hyp@opt: new	92
\glsdefmain: added hook to set toc title	12	\@gls@link: removed redundancy	94
4.07		renamed \@gls@type to \@gls@link	94
\@glossarysection: added optional argument when using unstarred version	37	\@gls@link@checkfirsthyper: new	94
\@gls@noidx@do: added \@global in case it's used in a tabular-like style	174	\@glsdisp: moved \@glsifhyper	111
\Acrfullplfmt: fixed no case change bug	196	moved check for first use to \@gls@link	111
\glsletentryfield: new	132	\@glspl@: moved \@glsifhyper	109
4.08		moved check for first use to \@gls@link	109
\@ACRlong: added \@do@gls@link@checkfirsthyper	326	\@ignored@glossaries: new ..	57
\@ACRshort: added \@do@gls@link@checkfirsthyper	325	General: added entrycounter option to printgloss family ..	178
\@Acrlong: added \@do@gls@link@checkfirsthyper	326	added \@nopostdot option to printgloss family	178
\@Acrshort: added \@do@gls@link@checkfirsthyper	325	added \@subentrycounter option to printgloss family	179
\@GLS@: moved \@glsifhyper ..	108	explicitly initialise hyper key ..	92
moved check for first use to \@gls@link	108	moved \@glsifhyper ...	126–132
\@GLSpl: moved \@glsifhyper ..	110	removed \@sACRlongpl	132
moved check for first use to \@gls@link	110	removed \@sAcrlongpl	131
\@GLS@: moved \@glsifhyper ..	107	removed \@sacrlongpl	130
moved check for first use to \@gls@link	107	removed \@sACRlong	130
\@GLSpl@: moved \@glsifhyper ..	109	removed \@sAcrlong	129
moved check for first use to \@gls@link	109	removed \@sacrlong	129
\@acrlong: added \@do@gls@link@checkfirsthyper	326	removed \@sACRshortpl ...	128
\@acrshort: added \@do@gls@link@checkfirsthyper	325	removed \@sAcrshortpl ...	127
\@closegls: new	149	removed \@sacrshortpl ...	127
\@gls@: moved \@glsifhyper ..	106	removed \@sACRshort	126
moved check for first use to \@gls@link	106	removed \@sAcrshort	126
\@gls@doautomake: new	25	removed \@sacrshort	125
\@gls@field@link: added as-signment of \@do@gls@link@checkfirsthyper	111	removed \@sgls@link	93
		removed \@sGLSdescplural ..	118
		removed \@sGlsdescplural ..	117
		removed \@sglsdescplural ..	117
		removed \@sGLSdesc	117
		removed \@sGlsdesc	116
		removed \@sglsdesc	116
		removed \@sglsdisp	111
		removed \@sGLSfirstplural ..	115
		removed \@sGlsfirstplural ..	115
		removed \@sglsfirstplural ..	114
		removed \@sGLSfirst	113
		removed \@sGlsfirst	113
		removed \@sglsfirst	113

removed \@sGLSname	116	removed spglspl	237
removed \@sGlsname	116	\ACRfull: removed \s@ACRfull	195
removed \@sglsname	115	switched to using \@gls@hyp@opt	
removed \@sGLSplural	114	195
removed \@sGlsplural	114	\Acrfull: removed \s@Acrfull	195
removed \@sglsplural	113	switched to using \@gls@hyp@opt	
removed \@sGLSpl	110	194
removed \@sGlspl	109	\acrfull: removed \s@acrfull	194
removed \@sglspl	108	switched to using \@gls@hyp@opt	
removed \@sGLSsymbolplural		194
.....	119	\ACRfullpl: removed \s@ACRfullpl	
removed \@sGLssymbolplural		196
.....	119	switched to using \@gls@hyp@opt	
removed \@sglssymbolplural		196
.....	119	\Acrfullpl: removed \s@Acrfullpl	
removed \@sGLSsymbol	119	196
removed \@sGLssymbol	118	switched to using \@gls@hyp@opt	
removed \@sglssymbol	118	196
removed \@sGLStext	112	\acrfullpl: removed \s@acrfullpl	
removed \@sGlstext	112	195
removed \@sglstext	112	switched to using \@gls@hyp@opt	
removed \@sGLSuseriii ...	122	195
removed \@sGlsuseriii ...	122	\ACRlong: switched to using	
removed \@sglsuseriii ...	122	\@gls@hyp@opt	130
removed \@sGLSuserii	121	\Acrlong: switched to using	
removed \@sGlsuserii	121	\@gls@hyp@opt	129
removed \@sglsuserii	121	\acrlong: switched to using	
removed \@sGLSuseriv	123	\@gls@hyp@opt	128
removed \@sGlsuseriv	123	\ACRlongpl: switched to using	
removed \@sglsuseriv	122	\@gls@hyp@opt	132
removed \@sGLSuseri	120	\Acrlongpl: switched to using	
removed \@sGlsuseri	120	\@gls@hyp@opt	131
removed \@sglsuseri	120	\acrlongpl: switched to using	
removed \@sGLSuservi	125	\@gls@hyp@opt	130
removed \@sGlsuservi	125	\ACRshort: switched to using	
removed \@sglsuservi	124	\@gls@hyp@opt	126
removed \@sGLSuserv	124	\Acrshort: switched to using	
removed \@sGlsuserv	124	\@gls@hyp@opt	126
removed \@sglsuserv	123	\acrshort: switched to using	
removed \@sGLS	108	\@gls@hyp@opt	125
removed \@sGls	107	\ACRshortpl: switched to using	
removed \@sgls	106	\@gls@hyp@opt	128
removed \@thirdofthree (de-		\Acrshortpl: switched to using	
finied in kernel)	105	\@gls@hyp@opt	127
removed sPGLS	239	\acrshortpl: switched to using	
removed sPgls	238	\@gls@hyp@opt	127
removed spgls	236	\forallacronyms: new	48
removed sPGLSpl	240	\GLS: switched to using	
removed sPglspl	238	\@gls@hyp@opt	107

\Gls: switched to using \@gls@hyp@opt 107	\glslink: switched to using \@gls@hyp@opt 93
\gls: switched to using \@gls@hyp@opt 106	\glslinkcheckfirsthyperhook: new 94
\gls@defglossaryentry: added check for ignored glossary ... 72	\glslinkvar: new 92
\gls@istfilebase: new 33	\GLSname: switched to using \@gls@hyp@opt 116
\glsaddkey: removed \@sGLS@user@⟨key⟩ 69	\Glsname: switched to using \@gls@hyp@opt 115
removed \@sGls@user@⟨key⟩ . 68	\glsname: switched to using \@gls@hyp@opt 115
removed \@sgls@user@⟨key⟩ . 68	\GLSpl: switched to using \@gls@hyp@opt 110
switched to using \@gls@hyp@opt 68, 69	\Glspl: switched to using \@gls@hyp@opt 109
\GLSdesc: switched to using \@gls@hyp@opt 117	\glspl: switched to using \@gls@hyp@opt 108
\Glsdesc: switched to using \@gls@hyp@opt 116	\GLSplural: switched to using \@gls@hyp@opt 114
\glsdesc: switched to using \@gls@hyp@opt 116	\Glsplural: switched to using \@gls@hyp@opt 114
\GLSdescplural: switched to us- ing \@gls@hyp@opt 118	\glsplural: switched to using \@gls@hyp@opt 113
\Glsdescplural: switched to us- ing \@gls@hyp@opt 117	\glsspace: new 194
\glsdescplural: switched to us- ing \@gls@hyp@opt 117	\GLSsymbol: switched to using \@gls@hyp@opt 118
\glsdisablehyper: added \KV@glslink@hyperfalse to definition 105	\Glsymbol: switched to using \@gls@hyp@opt 118
\glsdisp: switched to using \@gls@hyp@opt 111	\glssymbol: switched to using \@gls@hyp@opt 118
\glsdohyperlink: new 104	\GLSsymbolplural: switched to using \@gls@hyp@opt 119
\glsdohypertarget: new 104	\Glsymbolplural: switched to using \@gls@hyp@opt 119
\glsenablehyper: added \KV@glslink@hypertrue to definition 105	\glssymbolplural: switched to using \@gls@hyp@opt 119
\GLSfirst: switched to using \@gls@hyp@opt 113	\GLStext: switched to using \@gls@hyp@opt 112
\Glsfirst: switched to using \@gls@hyp@opt 113	\Glstext: switched to using \@gls@hyp@opt 112
\glsfirst: switched to using \@gls@hyp@opt 112	\glstext: switched to using \@gls@hyp@opt 112
\GLSfirstplural: switched to using \@gls@hyp@opt 115	\glstreenamfmt: new 280
\Glsfirstplural: switched to using \@gls@hyp@opt 115	\GLSuseri: switched to using \@gls@hyp@opt 120
\glsfirstplural: switched to using \@gls@hyp@opt 114	\Glsuseri: switched to using \@gls@hyp@opt 120
\glsifhyper: deprecated 92	\glsuseri: switched to using \@gls@hyp@opt 120

\GLSuserii: switched to using \@gls@hyp@opt	121	\glsuservi: switched to using \@gls@hyp@opt	124
\Glsuserii: switched to using \@gls@hyp@opt	121	\ifignoredglossary:new	57
\glsuserii: switched to using \@gls@hyp@opt	121	altlongragged4col: fixed bug that displayed description in- stead of symbol	261
\GLSuseriii: switched to using \@gls@hyp@opt	122	\newglossary: added starred ver- sion	54
\Glsuseriii: switched to using \@gls@hyp@opt	122	\newignoredglossary:new ...	57
\glsuseriii: switched to using \@gls@hyp@opt	121	\ns@newglossary: added \@glotype@<name>@log ...	55
\GLSuseriv: switched to using \@gls@hyp@opt	123	new	55
\Glsuseriv: switched to using \@gls@hyp@opt	123	\p@gls@hyp@opt:new	93
\glsuseriv: switched to using \@gls@hyp@opt	122	\PGLS: changed to use \@gls@hyp@opt	239
\GLSuserv: switched to using \@gls@hyp@opt	124	\Pgls: changed to use \@gls@hyp@opt	238
\Glsuserv: switched to using \@gls@hyp@opt	124	\pgls: changed to use \@gls@hyp@opt	236
\glsuserv: switched to using \@gls@hyp@opt	123	\PGLSpl: changed to use \@gls@hyp@opt	240
\GLSuservi: switched to using \@gls@hyp@opt	125	\Pglspl: changed to use \@gls@hyp@opt	238
\Glsuservi: switched to using \@gls@hyp@opt	124	\pglspl: changed to use \@gls@hyp@opt	237
		\s@gls@hyp@opt:new	92
		\s@newglossary:new	54
		automake:new	24

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	
\@@do@@wrglossary	160
\@@do@wrglossary	159
\@@glo@assign@sortkey	180
\@@glossarysec	5
\@@glossaryseclabel	6
\@@glossarysecstar	6
\@@gls@default@entryfmt	315
\@@gls@expand@field	63
\@@gls@fixbraces	162
\@@gls@noidx@nosanitizesort .	18
\@@gls@noidx@sanitizesort ...	18
\@@gls@nosanitizesort	18
\@@gls@sanitizesort	18
\@ACRlong	326
\@ACRshort	325
\@Acrlong	326
\@Acrshort	325
\@GLS@	108
\@GLSpl	110
\@Gls@	107
\@Gls@entry@field	133
\@Glspl@	109
\@PGLS	239
\@PGLS@	239
\@PGLSpl	240
\@PGLSpl@	240
\@Pgls	238
\@Pgls@	238
\@Pglspl	238
\@Pglspl@	239
\@acrfull	194
\@acrlong	326
\@acrshort	325
\@addtoacronymlists	14
\@closegls	149
\@delimN	191
\@delimR	190
\@disable@onlypremakeg	29
\@disable@premakecs	29
\@disabled@glsaddxdycounters	40
\@do@seeglossary	161
\@do@wrglossary	158, 289
\@glo@addchildren	168
\@glo@default@sorttype	9
\@glo@do@sortentries	169
\@glo@grabfirst	174
\@glo@no@assign@sortkey	180
\@glo@seeautonumberlist	7
\@glo@sortedinsert	169
\@glo@sortentries	168
\@glo@sorthandler@case	170
\@glo@sorthandler@letter ...	170
\@glo@sorthandler@nocase ...	170
\@glo@sorthandler@word	170
\@glo@sortmacro@case	171
\@glo@sortmacro@def	172
\@glo@sortmacro@def@do	172
\@glo@sortmacro@letter	171
\@glo@sortmacro@nocase	172
\@glo@sortmacro@standard ...	171
\@glo@sortmacro@use	173
\@glo@sortmacro@word	170
\@glo@storeentry	76
\@glo@types	53
\@glossary	157
\@glossary@default@style	6
\@glossaryentryfield	76
\@glossarysection	37
\@glossarysubentryfield	76
\@gls	106
\@gls@	106
\@gls@link	93
\@gls@access@display	313
\@gls@addpredefinedattributes	
.....	41
\@gls@assign@symbol@field ...	18
\@gls@assign@symbolplural@field	
.....	18
\@gls@checkactual	102
\@gls@checkbox	101
\@gls@checkescactual	100
\@gls@checkescbar	100
\@gls@checkesclevel	101
\@gls@checkescquote	99
\@gls@checklevel	102

<code>\@gls@checkmkidxchars</code>	98	<code>\@gls@saveentrycounter</code>	95
<code>\@gls@checkquote</code>	99	<code>\@gls@setacrstyle</code>	22
<code>\@gls@codepage</code>	47	<code>\@gls@setcounter</code>	56
<code>\@gls@counterwithin</code>	9	<code>\@gls@setupsort@def</code>	11
<code>\@gls@declareoption</code>	7	<code>\@gls@setupsort@standard</code>	10
<code>\@gls@default@value</code>	59	<code>\@gls@setupsort@use</code>	11
<code>\@gls@do@acronymsdef</code>	13	<code>\@gls@startswithexpandonce</code>	64
<code>\@gls@doautomake</code>	25	<code>\@gls@symbolsdef</code>	26
<code>\@gls@entry@field</code>	132	<code>\@gls@tmpb</code>	99
<code>\@gls@escbsdq</code>	97	<code>\@gls@toc</code>	38
<code>\@gls@expand@fields</code>	63	<code>\@gls@updatechecked</code>	99
<code>\@gls@fetchfield</code>	66	<code>\@gls@warnonglossdefined</code>	16
<code>\@gls@field@link</code>	111	<code>\@gls@warnontheGLOSSdefined</code>	16
<code>\@gls@fixbraces</code>	162	<code>\@gls@writedef</code>	65
<code>\@gls@forbidtext</code>	54	<code>\@gls@xdy@Lclass@Alpha-page-numbers</code>	43
<code>\@gls@getcounter</code>	56	<code>\@gls@xdy@Lclass@Appendix-page-numbers</code>	43
<code>\@gls@getcounterprefix</code>	161	<code>\@gls@xdy@Lclass@Roman-page-numbers</code>	43
<code>\@gls@getgrouptitle</code>	187	<code>\@gls@xdy@Lclass@alpha-page-numbers</code>	43
<code>\@gls@getothergrouptitle</code>	187	<code>\@gls@xdy@Lclass@arabic-page-numbers</code>	43
<code>\@gls@hyp@opt</code>	92	<code>\@gls@xdy@Lclass@arabic-section-numbers</code>	44
<code>\@gls@hypergroup</code>	245	<code>\@gls@xdy@Lclass@roman-page-numbers</code>	43
<code>\@gls@ifinlist</code>	39	<code>\@gls@xdy@locationlist</code>	42
<code>\@gls@indexdef</code>	26	<code>\@gls@xdy@checkbackslash</code>	103
<code>\@gls@keymap</code>	66, 234	<code>\@gls@xdy@checkquote</code>	103
<code>\@gls@link</code>	94	<code>\@gls@Alphacompositor</code>	33, 43
<code>\@gls@link@checkfirsthyper</code>	93	<code>\@gls@acronymlists</code>	14
<code>\@gls@loadlist</code>	8	<code>\@gls@defaultplural</code>	62
<code>\@gls@loadlong</code>	8	<code>\@gls@defaultsort</code>	62
<code>\@gls@loadsuper</code>	8	<code>\@gls@disp</code>	111
<code>\@gls@loadtree</code>	8	<code>\@gls@firstletter</code>	141
<code>\@gls@makefirsttuc</code>	241	<code>\@gls@hypernumber</code>	190
<code>\@gls@missingnumberlist</code>	62	<code>\@gls@link</code>	104
<code>\@gls@noaccess</code>	310	<code>\@gls@minrange</code>	142
<code>\@gls@noexpand@field</code>	62	<code>\@gls@nextpages</code>	180
<code>\@gls@noexpand@fields</code>	62	<code>\@gls@nodesc</code>	62
<code>\@gls@nohyperlist</code>	16	<code>\@gls@noname</code>	61
<code>\@gls@noidx@do</code>	174	<code>\@gls@nonextpages</code>	180
<code>\@gls@noidx@setsanitizesort</code>	20	<code>\@gls@openfile</code>	148
<code>\@gls@noref@warn</code>	155	<code>\@gls@order</code>	23
<code>\@gls@notranslatorhook</code>	21	<code>\@gls@spl@</code>	108
<code>\@gls@numbersdef</code>	26	<code>\@gls@target</code>	105
<code>\@gls@onlypremakeg</code>	28	<code>\@gls@widestname</code>	284
<code>\@gls@provide@newglossary</code>	53		
<code>\@gls@reference</code>	176		
<code>\@gls@renewglossary</code>	157		
<code>\@gls@sanitizedesc</code>	17		
<code>\@gls@sanitizename</code>	17		
<code>\@gls@sanitizesort</code>	18		
<code>\@gls@sanitizesymbol</code>	18		

\@glswritefiles	156	\acl	209
\@ignored@glossaries	57	\Aclp	209
\@istfilename	33	\aclp	209
\@makeglossary	148	\Acp	210
\@mfu@nocaplist	243	\acp	210
\@newglossary	56	\acrfootnote	212
\@newglossaryentryposthook ..	76	\ACRfull	195
\@newglossaryentryprehook ...	76	\Acrfull	194
\@no@makeglossaries	155	\acrfull	194, 198, 207, 335
\@no@post@desc	32	\ACRfullfmt	195
\@nopostdesc	32	\Acrfullfmt	195
\@onlypremakeg	28	\acrfullfmt	194
\@p@glossarysection	37	\acrfullformat	87, 194
\@pgls	237	\ACRfullpl	196
\@pgls@	237	\Acrfullpl	196
\@pglspl	237	\acrfullpl	195
\@pglspl@	237	\ACRfullplfmt	196
\@print@glossary	166	\Acrfullplfmt	196
\@print@noidx@glossary	173	\acrfullplfmt	196
\@printgloss@setsort	164	\acrlinkfootnote	212
\@printglossary	165	\acrlinkfullformat	194
\@set@glo@numformat	97, 289	\ACRlong	130
\@wrglossary	157	\Acrlong	129
\@xdy@main@language	24	\acrlong	128
\@xdy@attributelist	39	\ACRlongpl	132
\@xdy@attributes	39	\Acrlongpl	131
\@xdy@language	46	\acrlongpl	130
\@xdy@lettergroups	47	\acrnameformat	197, 218
\@xdy@locationclassorder	45	\acrno linkfootnote	212
\@xdy@locref	39	acronym (option)	13
\@xdy@requiredstyles	46	acronym styles:	
\@xdy@sortrules	45	dua	204, 331
\@xdy@useralphabets	42	dua-desc	206, 334
\@xdy@userlocationdefs	44	footnote	206, 334
\@xdy@userlocationnames	44	footnote-desc	208, 336
A			
\Ac	210	footnote-sc	207, 336
\ac	210	footnote-sc-desc	208, 337
access (key)	308	footnote-sm	207, 336
accsupp package	307	footnote-sm-desc	208, 337
\accsuppglossaryentryfield .	328	long-sc-short	201
\accsuppglossarysubentryfield		long-sc-short-desc ..	202, 330
.....	328	long-short	200, 328
\Acf	210	long-short-desc	202, 330
\acf	209	long-sm-short	201
\Acfp	210	long-sm-short-desc ..	203, 330
\acfp	210	sc-short-long	202
\Acl	209	sc-short-long-desc ..	203, 331
		short-long	200, 329
		short-long-desc	203, 331

sm-short-long 202
 sm-short-long-desc .. 203, 331
 \acronymentry 198
 \acronymfont
 87, 197, 214, 218, 221, 224
 acronymlists (option) 15
 \acronymname 29
 acronyms (option) 14
 \acronymsort 198
 \acronymtype 13, 193
 \acrpluralsuffix 193
 \ACRshort 126
 \Acrshort 126
 \acrshort 125
 \ACRshortpl 128
 \Acrshortpl 127
 \acrshortpl 127
 \Acs 209
 \acs 209
 \Acsp 209
 \acsp 209
 \addglossarytocaptions 30
 \addto 30
 align (environment) 78, 95, 96
 altlist (style) 250
 altlistgroup (style) 250
 altlisthypergroup (style) 250
 altlong4col (style) 256
 altlong4colborder (style) 257
 altlong4colheader (style) 257
 altlong4colheaderborder (style) 257
 altlongragged4col (style) 261
 altlongragged4colborder (style) 262
 altlongragged4colheader (style) 262
 altlongragged4colheaderborder
 (style) 263
 \altnewglossary 56
 altsuper4col (style) 272
 altsuper4colborder (style) 273
 altsuper4colheader (style) 273
 altsuper4colheaderborder
 (style) 273
 altsuperragged4col (style) 278
 altsuperragged4colborder
 (style) 279
 altsuperragged4colheader
 (style) 278
 altsuperragged4colheaderborder
 (style) 279

alttree (style) 284
 alttreegroup (style) 287
 alttreehypergroup (style) 287
 amsgen package 4, 91
 amsmath package 78
 \andname 30
 array package 258, 274
 article class 161
 automake (option) 24

B

babel package 28, 29, 31, 46, 344

C

\capitalisewords 242
 \changes 5
 \compatglossarystyle 294
 compatible-2.07 (option) 25
 compatible-3.07 (option) 25
 \compatibleglossentry .. 183, 307
 \compatiblesubglossentry 185, 308
 counter (key) 59
 counter (option) 15
 \CustomAcronymFields 226
 \CustomNewAcronymDef 227

D

datatool package 169
 \DeclareAcronymList 14
 \DefaultNewAcronymDef .. 211, 337
 \defentryfmt 91
 \defglstdisplay 90
 \defglstdisplayfirst 90
 \defglssentry 55
 \defglssentryfmt 54, 58, 59, 82
 \DefineAcronymSynonyms 209
 \delimN 35, 190
 \delimR 35, 190
 description (environment) 248, 249
 description (key) 58
 description (option) 22
 descriptionaccess (key) 309
 \DescriptionDUANewAcronymDef 215
 \DescriptionFootnoteNewAcronymDef
 213, 339
 \descriptionname 30
 \DescriptionNewAcronymDef ..
 217, 339
 descriptionplural (key) 58
 descriptionpluralaccess (key) 309

\detokenize 49
doc package 4, 5, 12
dua (acrstyle) 204, 331
dua (option) 23
dua-desc (acrstyle) 206, 334
\DUANewAcronymDef 224

E

entrycounter (option) 9
entrycounterwithin (option) 9
\entryname 29
environments:
 align 78, 95, 96
 description 248, 249
 longtable 8, 228, 251–263
 multicols 263
 supertabular ... 8, 228, 267–279
 theglossary 5,
 16, 35, 183, 189, 265, 266, 282–285
 theindex 280
equation (counter) 95, 96
etoolbox package 4, 240

F

file types
 .aux 166
 .glo 76
 .ist 141, 147, 148
 .toc 38
 .xdy 33
 glo 234
\findrootlanguage 46
first (key) 59
firstaccess (key) 309
\firstacronymfont 87, 197
firstplural (key) 59
firstpluralaccess (key) 309
footnote (acrstyle) 206, 334
footnote (option) 22
footnote-desc (acrstyle) .. 208, 336
footnote-sc (acrstyle) 207, 336
footnote-sc-desc (acrstyle) 208, 337
footnote-sm (acrstyle) 207, 336
footnote-sm-desc (acrstyle) 208, 337
\FootnoteNewAcronymDef . 219, 340
\forallacronyms 48
\forallglossaries 48
\forallglsentries 48
\forglsentries 48

G

garamondx package 193
\Genacrfullformat 89, 324
\genacrfullformat 89, 324
\GenericAcronymFields 198
\Genplacrfullformat 90, 324
\genplacrfullformat 89, 324
\glo@grabfirst 173
\glolinkprefix 95
glossareentry (counter) 181
glossaries package 26,
 46, 47, 142, 228, 234, 248, 287, 307
glossaries-accsupp package ... 76, 307
\GlossariesWarning 16
\GlossariesWarningNoLine 16
\glossary 54, 148, 157, 188
glossary counters:
 glossaryentry 180
 glossarysubentry 181
glossary keys:
 access 308
 counter 59
 description 58
 descriptionaccess 309
 descriptionplural 58
 descriptionpluralaccess . 309
 first 59
 firstaccess 309
 firstplural 59
 firstpluralaccess 309
 long 61
 longaccess 309
 longplural 61
 longpluralaccess 310
 name 58
 nonumberlist 60
 parent 60
 plural 58
 pluralaccess 309
 see 60
 short 61
 shortaccess 309
 shortplural 61
 shortpluralaccess 309
 sort 58
 symbol 59
 symbolaccess 309
 symbolplural 59
 symbolpluralaccess 309

text	58
textaccess	308
type	59
user1	60
user2	60
user3	61
user4	61
user5	61
user6	61
glossary package	1, 192
glossary styles:	
altlist	250, 295
altlist	250
altlistgroup	250, 296
altlistgroup	250
altlisthypergroup ...	250, 296
altlisthypergroup	250
altlong4col ..	256, 257, 261, 298
altlong4col	256
altlong4colborder ...	257, 298
altlong4colborder	257
altlong4colheader ...	257, 298
altlong4colheader	257
altlong4colheaderborder .	
.....	257, 298
altlong4colheaderborder .	257
altlongragged4col	261, 262, 299
altlongragged4col	261
altlongragged4colborder .	
.....	262, 299
altlongragged4colborder .	262
altlongragged4colheader .	
.....	262, 299
altlongragged4colheader .	262
altlongragged4colheaderborder	
.....	263, 299
altlongragged4colheaderborder	
.....	263
altsuper4col .	272, 273, 278, 307
altsuper4col	272
altsuper4colborder ..	273, 307
altsuper4colborder	273
altsuper4colheader ..	273, 307
altsuper4colheader	273
altsuper4colheaderborder	
.....	273, 307
altsuper4colheaderborder	273
altsuperragged4col	278, 279, 305
altsuperragged4col	278
altsuperragged4colborder	
.....	279, 305
altsuperragged4colborder	279
altsuperragged4colheader	
.....	278, 305
altsuperragged4colheader	278
altsuperragged4colheaderborder	
.....	279, 305
altsuperragged4colheaderborder	
.....	279
alttree	266, 284, 287, 301
alttree	284
alttreegroup	287, 303
alttreegroup	287
alttreehypergroup ...	287, 303
alttreehypergroup	287
index	263, 280, 281, 299
index	280
indexgroup	281, 300
indexgroup	281
indexhypergroup	281, 300
indexhypergroup	281
inline	294
inline	246
list	6, 248–251, 295
list	248
listdotted	251, 296
listdotted	251
listgroup	249, 295
listgroup	249
listhypergroup	249, 295
listhypergroup	249
long	252, 253, 258, 296, 298
long	252
long3col	253, 254, 297
long3col	253
long3colborder	254, 297
long3colborder	254
long3colheader	254, 297
long3colheader	254
long3colheaderborder	254, 297
long3colheaderborder	254
long4col	255, 256, 297
long4col	255
long4colborder	256, 297
long4colborder	256
long4colheader	255, 297
long4colheader	255
long4colheaderborder	256, 297

long4colheaderborder	256
longborder	252, 296
longborder	252
longheader	253, 296
longheader	253
longheaderborder	253, 296
longheaderborder	253
longragged	258–260
longragged	258
longragged3col	...	260, 261, 298
longragged3col	260
longragged3colborder		260, 299
longragged3colborder	260
longragged3colheader		261, 299
longragged3colheader	261
longragged3colheaderborder	261, 299
longragged3colheaderborder	261
longraggedborder	259, 298
longraggedborder	259
longraggedheader	259, 298
longraggedheader	259
longraggedheaderborder		259, 298
longraggedheaderborder	..	259
mcolalttree	266, 304
mcolalttree	266
mcolalttreegroup	266, 304
mcolalttreegroup	266
mcolalttreehypergroup		266, 304
mcolalttreehypergroup	...	266
mcolindex	264, 303
mcolindex	263
mcolindexgroup	264, 303
mcolindexgroup	264
mcolindexhypergroup	.	264, 303
mcolindexhypergroup	264
mcoltree	264, 303
mcoltree	264
mcoltreegroup	303
mcoltreegroup	264
mcoltreehypergroup	..	265, 303
mcoltreehypergroup	265
mcoltreenoname	265, 303
mcoltreenoname	265
mcoltreenonamegroup	.	265, 303
mcoltreenonamegroup	265
mcoltreenonamehypergroup	265, 303
mcoltreenonamehypergroup		265
sublistdotted	296
sublistdotted	251
super	267–269, 275, 305
super	267
super3col	269, 270, 306
super3col	269
super3colborder	270, 306
super3colborder	270
super3colheader	270, 306
super3colheader	270
super3colheaderborder		270, 306
super3colheaderborder	...	270
super4col	271, 272, 306
super4col	271
super4colborder	272, 307
super4colborder	272
super4colheader	271, 306
super4colheader	271
super4colheaderborder		272, 307
super4colheaderborder	...	272
superborder	268, 305
superborder	268
superheader	268, 306
superheader	268
superheaderborder	...	269, 306
superheaderborder	269
superragged	274, 276, 304
superragged	274
superragged3col	..	276, 277, 304
superragged3col	276
superragged3colborder		277, 305
superragged3colborder	...	277
superragged3colheader		277, 305
superragged3colheader	...	277
superragged3colheaderborder	277, 305
superraggedborder	...	275, 304
superraggedborder	275
superraggedheader	...	275, 304
superraggedheader	275
superraggedheaderborder	.	
superraggedheaderborder	276, 304
superraggedheaderborder	.	276
superraggedright3colheaderborder	277
tree	264, 281–284, 300
tree	281
treegroup	265, 282, 301

treegroup	282	\gls@assign@desc	69
treehypergroup	282, 301	\gls@assign@desc@field	17
treehypergroup	282	\gls@assign@descplural@field	17
treenoname ...	265, 283, 284, 301	\gls@assign@field	64
treenoname	283	\gls@assign@name@field	17
treenonamegroup	284, 301	\gls@assign@type@field	17
treenonamegroup	284	\gls@checkisacronymlist	15
treenonamehypergroup	284, 301	\gls@checkseeallowed	60
treenonamehypergroup	284	\gls@codepage	24
glossary-hypernav package	141	\gls@defglossaryentry	70
glossary-list package	6, 8, 248	\gls@disablepagerefexpansion	158
glossary-long package		\gls@docclearpage	38
.....	8, 251, 252, 261, 267	\gls@doentryfmt	54
glossary-longragged package	258	\gls@hypergroup prerun	245
glossary-mcols package	263	\gls@ifnotmeasuring	78
glossary-super package		\gls@istfilebase	33
.....	8, 252, 267, 274, 278	\gls@level	62
glossary-superragged package	274	\gls@noidxglossary	156
glossary-tree package	8, 280	\gls@numberpage	158
glossaryentry (counter) ..	9, 181, 182	\gls@protected@pagefmts	158
glossaryentry (counter)	180	\gls@Romanpage	159
\glossaryentryfield	185, 189	\gls@romanpage	159
\glossaryentrynumber	180	\gls@save@numberlist	163
\glossaryentrynumbers		\gls@suffixF	34
.....	7, 34, 165, 166	\gls@suffixFF	34
\glossaryheader	183, 189	\glsaccessdisplay	315
\glossarymark	36	\glsaccsupp	313
\glossaryname	29, 30	\glsadd	80, 139, 140, 188
\glossarypostamble	36, 189	\glsadd options	
\glossarypreamble	35, 189	counter	140
\glossarysection	6, 36, 54	format	140, 189
\glossarystyle	188, 228	\glsaddall	49, 80, 140
glossarysubentry (counter) ...		\glsaddall options	
.....	9, 181, 182	types	140
glossarysubentry (counter) ...	181	\glsaddallunused	140
\glossarysubentryfield	185	\glsaddkey	67
\glossentry	59, 183	\GlsAddLetterGroup	47
\Glossentrydesc	184	\GlsAddSortRule	45
\glossentrydesc	184, 327	\GlsAddXdyAlphabet	42
\Glossentryname	184	\GlsAddXdyAttribute	40, 287
\glossentryname	184, 327	\GlsAddXdyCounters	39, 288
\Glossentrysymbol	184	\GlsAddXdyLocation	44, 288
\glossentrysymbol	184, 327	\GlsAddXdyStyle	46
\GLS	107	\glsautoprefix	6
\Gls	107, 109, 240	\glsclearpage	38
\gls	4, 59,	\gls closebrace	141
	80, 92, 106–108, 112–124, 182, 236	\glscompositor	33, 44
\gls@Alphpage	158	\glscounter	15, 56
\gls@alphpage	158	\GlsDeclareNoHyperList	16

<code>\glsdefaulttype</code>	13	<code>\Glsentrylongpl</code>	137
<code>\glsdefmain</code>	12	<code>\glsentrylongpl</code>	137
<code>\GLSdesc</code>	117	<code>\glsentrylongpluralaccess</code> ..	313
<code>\Glsdesc</code>	116	<code>\Glsentryname</code>	133
<code>\glsdesc</code>	116, 117	<code>\glsentryname</code>	133, 163
<code>\GLSdescplural</code>	118	<code>\glsentrynumberlist</code>	138, 153
<code>\Glsdescplural</code>	117	<code>\Glsentryplural</code>	134
<code>\glsdescplural</code>	117, 118	<code>\glsentryplural</code>	134
<code>\glsdescriptionaccessdisplay</code>	314	<code>\glsentrypluralaccess</code>	312
<code>\glsdescriptionpluralaccessdisplay</code>	314	<code>\Glsentryprefix</code>	236
.....	314	<code>\glsentryprefix</code>	235
<code>\glsdescwidth</code> ...	252, 258, 267, 274	<code>\Glsentryprefixfirst</code>	235
<code>\glsdetoklabel</code>	49	<code>\glsentryprefixfirst</code>	235
<code>\glsdisablehyper</code>	105	<code>\Glsentryprefixfirstplural</code> .	235
<code>\glsdisp</code>	111	<code>\glsentryprefixfirstplural</code> .	235
<code>\glsdisplay</code>	81, 90, 106	<code>\Glsentryprefixplural</code>	236
<code>\glsdisplayfirst</code>	81, 90, 106	<code>\glsentryprefixplural</code>	235
<code>\glsdisplaynumberlist</code>	138, 154, 176	<code>\Glsentryshort</code>	137
<code>\glsdohyperlink</code>	104	<code>\glsentryshort</code>	137
<code>\glsdohypertarget</code>	104	<code>\glsentryshortaccess</code>	313
<code>\glsdoifexists</code>	50	<code>\Glsentryshortpl</code>	137
<code>\glsdoifexistsorwarn</code>	50	<code>\glsentryshortpl</code>	137
<code>\glsdoifnoexists</code>	50	<code>\glsentryshortpluralaccess</code> .	313
<code>\glsdoparenifnotempty</code>	221	<code>\glsentrysort</code>	135
<code>\glsenablehyper</code>	105	<code>\Glsentrysymbol</code>	134
<code>\glsentryaccess</code>	312	<code>\glsentrysymbol</code>	134
<code>\glsentrycounter</code>	95	<code>\glsentrysymbolaccess</code>	312
<code>\glsentrycounterlabel</code>	182	<code>\Glsentrysymbolplural</code>	135
<code>\Glsentrydesc</code>	133	<code>\glsentrysymbolplural</code>	134
<code>\glsentrydesc</code>	133	<code>\glsentrysymbolpluralaccess</code>	312
<code>\glsentrydescaccess</code>	312	<code>\Glsentrytext</code>	134
<code>\Glsentrydescplural</code>	134	<code>\glsentrytext</code>	49, 134, 163
<code>\glsentrydescplural</code>	133	<code>\glsentrytextaccess</code>	312
<code>\glsentrydescpluralaccess</code> ..	312	<code>\glsentrytype</code>	135
<code>\Glsentryfirst</code>	135	<code>\Glsentryuseri</code>	136
<code>\glsentryfirst</code>	135	<code>\glsentryuseri</code>	135
<code>\glsentryfirstaccess</code>	312	<code>\Glsentryuserii</code>	136
<code>\Glsentryfirstplural</code>	135	<code>\glsentryuserii</code>	136
<code>\glsentryfirstplural</code>	135	<code>\Glsentryuseriii</code>	136
<code>\glsentryfirstpluralaccess</code> .	312	<code>\glsentryuseriii</code>	136
<code>\glsentryfmt</code>	58, 59, 81	<code>\Glsentryuseriv</code>	136
<code>\Glsentryfull</code>	138	<code>\glsentryuseriv</code>	136
<code>\glsentryfull</code> ...	138, 198, 207, 336	<code>\Glsentryuserv</code>	136
<code>\Glsentryfullpl</code>	138	<code>\glsentryuserv</code>	136
<code>\glsentryfullpl</code>	138	<code>\Glsentryuservi</code>	137
<code>\glsentryitem</code>	182	<code>\glsentryuservi</code>	137
<code>\Glsentrylong</code>	137	<code>\glsexpandfields</code>	64
<code>\glsentrylong</code>	137	<code>\GLSfirst</code>	113
<code>\glsentrylongaccess</code>	313	<code>\Glsfirst</code>	113

<code>\glsfirst</code>	112, 113	<code>\glslongpluralaccessdisplay</code>	315
<code>\glsfirstaccessdisplay</code>	314	<code>\glslongpluralaccesskey</code>	342
<code>\GLSfirstplural</code>	115	<code>\glslongpluralkey</code>	194
<code>\Glsfirstplural</code>	115	<code>\glslongtok</code>	197
<code>\glsfirstplural</code>	114, 115	<code>\glsmakefirstuc</code>	242
<code>\glsfirstpluralaccessdisplay</code>	314	<code>\glsncols</code>	263
<code>\glsgenacfmt</code>	87, 322	<code>\glsmoveentry</code>	76
<code>\glsgenentryfmt</code>	85, 319	<code>\GLSname</code>	116
<code>\glsgetgrouplabel</code>	188	<code>\Glsname</code>	115
<code>\glsgetgrouptitle</code>	141, 187	<code>\glsname</code>	115, 116
<code>\glsglossarymark</code>	9, 36	<code>\glsnameaccessdisplay</code>	313
<code>\glsgroupheading</code>	186, 189	<code>\glsnamefont</code>	189, 280
<code>\glsgroupskip</code>	186, 189, 248	<code>\glsnavhyperlink</code>	244
<code>\glshyperlink</code>	139	<code>\glsnavhypertarget</code>	244
<code>\glshypernavsep</code>	246	<code>\glsnavigation</code>	245
<code>\glshypernumber</code>	34, 190	<code>\glsnextpages</code>	180
<code>\glsifhyper</code>	92	<code>\glsnoexpandfields</code>	64
<code>\glsifhyperon</code>	92	<code>\glsnoidxdisplayloc</code>	176
<code>\glsIfListOfAcronyms</code>	14	<code>\glsnoidxdisplaylocclsthander</code>	
<code>\glsinlinedescformat</code>	248	176
<code>\glsinlinedopostchild</code> ..	246, 247	<code>\glsnoidxlocclsthander</code>	175
<code>\glsinlineemptydescformat</code> ..	248	<code>\glsnoidxlocclsthander</code>	176
<code>\glsinlinenameformat</code>	248	<code>\glsnoidxnumberlistloopclsthander</code>	
<code>\glsinlineparentchildseparator</code>		155
.....	248	<code>\glsnoidxstripaccents</code>	19
<code>\glsinlinepostchild</code>	248	<code>\glsnonextpages</code>	180
<code>\glsinlineseparator</code>	248	<code>\glsnoxindywarning</code>	39
<code>\glsinlinesubdescformat</code>	248	<code>\glsnumberformat</code>	34
<code>\glsinlinesubnameformat</code>	248	<code>\glsnumberlistloop</code>	155
<code>\glsinlinesubseparator</code>	248	<code>\glsnumbersgroupname</code> .	30, 141, 186
<code>\glskeylisttok</code>	197	<code>\glsnumlistlastsep</code>	139
<code>\glslabeltok</code>	197	<code>\glsnumlistsep</code>	139
<code>\glsletentryfield</code>	132	<code>\glsopenbrace</code>	141
<code>\glslink</code> 80, 81, 93, 106, 139, 188, 189		<code>\glsorder</code>	23
<code>\glslink options</code>		<code>\glsorg@endtheglossary</code>	5
counter	91, 106, 234	<code>\glsorg@glossary</code>	4
format	91, 106, 189	<code>\glsorg@theglossary</code>	5
hyper	92, 93, 106	<code>\glsorg@wrglossary</code>	4
local	92	<code>\glspagelistwidth</code> 252, 258, 267, 274	
<code>\glslinkcheckfirsthyperhook</code> .	94	<code>\glspar</code>	32
<code>\glslinkvar</code>	92	<code>\GLSpl</code>	110
<code>\glslistdottedwidth</code>	251	<code>\Glspl</code>	109, 240
<code>\glslocalreset</code>	78	<code>\glspl</code>	80, 108–110
<code>\glslocalresetall</code>	79	<code>\GLSplural</code>	114
<code>\glslocalunset</code>	79	<code>\Glsplural</code>	114
<code>\glslocalunsetall</code>	80	<code>\glsplural</code>	113, 114
<code>\glslongaccessdisplay</code>	315	<code>\glspluralaccessdisplay</code>	314
<code>\glslongaccesskey</code>	342	<code>\glspluralsuffix</code>	30, 58, 59
<code>\glslongkey</code>	194	<code>\glspostdescription</code>	8

<code>\glspostinline</code>	248	<code>\Glssymbolplural</code>	119
<code>\glsprestandardsort</code>	10	<code>\glssymbolplural</code>	119
<code>\glsquote</code>	141	<code>\glssymbolpluralaccessdisplay</code>	314
<code>\glsrefentry</code>	182	<code>\glssymbolsgroupname</code>	30, 141, 186
<code>\glsreset</code>	78	<code>\glstarget</code>	183
<code>\glsresetall</code>	79	<code>\GLStext</code>	112
<code>\glsresetentrylist</code>	180	<code>\Glstext</code>	112
<code>\glsresetsubentrycounter</code>	181	<code>\glstext</code>	112
<code>\glsee</code>	162	<code>\glstextaccessdisplay</code>	314
<code>\glseeformat</code>	144, 162	<code>\glstextformat</code>	81
<code>\glseeitem</code>	163	<code>\glstextup</code>	193
<code>\glseeitemformat</code>	163	<code>\glstreeindent</code>	282, 283
<code>\glseelastsep</code>	163	<code>\glstreenamefmt</code>	280
<code>\glseelist</code>	162	<code>\glsunset</code>	79
<code>\glseesep</code>	163	<code>\glsunsetall</code>	80
<code>\glsSetAlphaCompositor</code>	34	<code>\GlsUseAcrEntryDispStyle</code>	200
<code>\glsSetCompositor</code>	33	<code>\GlsUseAcrStyleDefs</code>	200
<code>\glssetexpandfield</code>	17	<code>\GLSuseri</code>	120
<code>\glssetnoexpandfield</code>	17	<code>\Glsuseri</code>	120
<code>\glsSetSuffixF</code>	34	<code>\glsuseri</code>	120
<code>\glsSetSuffixFF</code>	34	<code>\GLSuserii</code>	121
<code>\glssettoctitle</code>	29	<code>\Glsuserii</code>	121
<code>\glssetwidest</code>	284	<code>\glsuserii</code>	121
<code>\GlsSetXdyCodePage</code>	47	<code>\GLSuseriii</code>	122
<code>\GlsSetXdyFirstLetterAfterDigits</code>	141	<code>\Glsuseriii</code>	122
<code>\GlsSetXdyLanguage</code>	46	<code>\glsuseriii</code>	121, 122
<code>\GlsSetXdyLocationClassOrder</code>	45	<code>\GLSuseriv</code>	123
<code>\GlsSetXdyMinRangeLength</code>	142	<code>\Glsuseriv</code>	123
<code>\GlsSetXdyStyles</code>	46	<code>\glsuseriv</code>	122, 123
<code>\glsshortaccessdisplay</code>	314	<code>\GLSuserv</code>	124
<code>\glsshortaccesskey</code>	342	<code>\Glsuserv</code>	124
<code>\glsshortkey</code>	193	<code>\glsuserv</code>	123, 124
<code>\glsshortpluralaccessdisplay</code>	314	<code>\GLSuservi</code>	125
<code>\glsshortpluralaccesskey</code>	342	<code>\Glsuservi</code>	124
<code>\glsshortpluralkey</code>	194	<code>\glsuservi</code>	124, 125
<code>\glsshorttok</code>	197	<code>\glswrite</code>	151
<code>\glssortnumberfmt</code>	10	<code>\glswritedefhook</code>	69
<code>\glsspace</code>	194	<code>\gMFUnocap</code>	243
<code>\glsstepentry</code>	181		
<code>\glsstepsubentry</code>	181		
<code>\glssubentrycounterlabel</code>	182		
<code>\glssubentryitem</code>	182		
<code>\GLSsymbol</code>	118		
<code>\Glssymbol</code>	118		
<code>\glssymbol</code>	118		
<code>\glssymbolaccessdisplay</code>	314		
<code>\glssymbolnav</code>	246		
<code>\GLSsymbolplural</code>	119		

H

<code>\hyperbf</code>	192
<code>\hyperemph</code>	192
<code>hyperfirst (option)</code>	22
<code>\hyperit</code>	192
<code>\hyperlink</code>	104
<code>\hypermd</code>	192
<code>\hyperpage</code>	190
<code>hyperref package</code>	161, 164, 190, 234

`\hyperm` 191
`\hypersc` 192
`\hypersf` 191
`\hypersl` 192
`\hypertarget` 105
`\hypertt` 191
`\hyperup` 192

I

`\if@gls@docloaded` 4
`\if@gls@isacronymlist` 15
`\ifglossaryexists` 49
`\ifglscdescsuppressed` 51
`\ifglsentryexists` 49
`\ifglshaschildren` 50
`\ifglshasdesc` 51
`\ifglshasfield` 52
`\ifglshaslong` 51
`\ifglshasparent` 51
`\ifglshasprefix` 236
`\ifglshasprefixfirst` 236
`\ifglshasprefixfirstplural` 236
`\ifglshasprefixplural` 236
`\ifglshasshort` 52
`\ifglshassymbol` 51
`\ifglstranslate` 21
`\ifglxsused` 49, 78
`\ifglxsindy` 23
`\ifignoredglossary` 57
`index (option)` 26
`index (style)` 280
`indexgroup (style)` 281
`indexhypergroup (style)` 281
`indexonlyfirst (option)` 22
`inline (style)` 246
`\inputencodingname` 24
`\istfile` 156
`\istfilename` 33
`\item` 189, 248, 280

L

`link text` 81
`list (style)` 248
`listdotted (style)` 251
`listgroup (style)` 249
`listhypergroup (style)` 249
`\loadglsentries` 12, 80
`long (key)` 61
`long (style)` 252
`long-sc-short (acrstyle)` 201

`long-sc-short-desc (acrstyle)` ..
 202, 330
`long-short (acrstyle)` 200, 328
`long-short-desc (acrstyle)` . 202, 330
`long-sm-short (acrstyle)` 201
`long-sm-short-desc (acrstyle)` ..
 203, 330
`long3col (style)` 253
`long3colborder (style)` 254
`long3colheader (style)` 254
`long3colheaderborder (style)` .. 254
`long4col (style)` 255
`long4colborder (style)` 256
`long4colheader (style)` 255
`long4colheaderborder (style)` .. 256
`longaccess (key)` 309
`longborder (style)` 252
`longheader (style)` 253
`longheaderborder (style)` 253
`\longnewglossaryentry` 58, 69
`longplural (key)` 61
`longpluralaccess (key)` 310
`\longprovideglossaryentry` ... 70
`longragged (style)` 258
`longragged3col (style)` 260
`longragged3colborder (style)` .. 260
`longragged3colheader (style)` .. 261
`longragged3colheaderborder`
`(style)` 261
`longraggedborder (style)` 259
`longraggedheader (style)` 259
`longraggedheaderborder (style)` 259
`longtable (environment)`
 8, 228, 251–263
`longtable package` 251, 258

M

`\makefirstuc` 240
`makeglossaries`
 23, 33, 46, 47, 54, 150, 166
`\makeglossaries`
 ... 24, 28, 32–34, 53, 56, 150, 152
`\makeglossary` 152
`makeindex` 357
`makeindex` 10, 23, 24, 30,
 33–35, 54, 56, 58, 77, 97, 100,
 141, 144, 146, 148, 160, 186, 289
`delim_n` 34
`delim_r` 35

page_compositor 33
 special characters 98, 99, 141
 makeindex (option) 23
 \makenoidxglossaries 20, 152
 mcolalmtree (style) 266
 mcolalmtreegroup (style) 266
 mcolalmtreehypergroup (style) . 266
 mcolindex (style) 263
 mcolindexgroup (style) 264
 mcolindexhypergroup (style) ... 264
 mcoltree (style) 264
 mcoltreegroup (style) 264
 mcoltreehypergroup (style) 265
 mcoltreename (style) 265
 mcoltreenamegroup (style) ... 265
 mcoltreenamehypergroup
 (style) 265
 memoir class 157
 mfirstuc package 1, 243
 \mfirstucMakeUppercase 242
 \mfu@checkword 242
 \MFUclear 243
 \MFUnocap 243
 multicol package 263
 multicol (environment) 263

N

name (key) 58
 \new@glossaryentry 65
 \newacronym 22, 23, 61, 80, 81, 192, 193
 \newacronymhook 197
 \newacronymstyle 199
 \newglossary 15, 54, 56, 148, 150, 177
 \newglossaryentry 58, 64, 80, 81, 193
 \newglossaryentry options
 access 310, 312
 counter 59
 description 22, 57, 58,
 62, 64, 70, 116, 133, 193, 220, 309
 descriptionaccess 312, 314
 descriptionplural 117, 309
 descriptionpluralaccess .. 312, 314
 first 59,
 73, 105, 112, 135, 218, 223, 309
 firstaccess 312, 314
 firstplural 59, 114, 135, 309
 firstpluralaccess 312, 314
 format 143
 long 87, 137, 309

longaccess 313, 315
 longplural 137, 310
 longpluralaccess 313, 315
 name 57,
 58, 61, 64, 70, 115, 133, 163, 308
 nonumberlist 60
 parent 60, 64
 plural 58, 73, 113, 309
 pluralaccess 312, 314
 prefix 235
 prefixfirst 235
 prefixfirstplural 235
 prefixplural 235
 see 7, 60, 151, 152
 short 87, 137, 309
 shortaccess 313, 314
 shortplural 137, 309
 shortpluralaccess 313, 314
 sort 58, 135, 186
 symbol 58, 59, 118, 214,
 216, 218, 223, 255, 271, 308–310
 symbolaccess 312, 314
 symbolplural 119, 309
 symbolpluralaccess 312, 314
 text 58,
 59, 105, 112, 134, 214, 218, 308
 textaccess 312, 314
 type 12, 59, 80, 135
 user1 120, 135, 310
 user2 121, 136
 user3 121, 136
 user4 122, 136
 user5 123, 136
 user6 124, 137, 310
 \newglossarystyle 189
 \newignoredglossary 57
 nogroupskip (option) 9
 nohypertypes (option) 16
 \noist 148, 234, 294
 nolist (option) 8
 nolong (option) 8
 nomain (option) 13
 nonumberlist (key) 60
 nonumberlist (option) 7
 \nopostdesc 32
 nopostdot (option) 9
 noredefwarn (option) 16
 nostyles (option) 8
 nosuper (option) 8

notranslate (option)	21
notree (option)	8
nowarn (option)	16
\ns@newglossary	55
numberedsection (option)	6
numberline (option)	5
numbers (option)	26

O

\oldacronym	192
order (option)	23

P

\p@gl@s@hyp@opt	93
package options:	
acronym	13, 29, 164, 193
true	14
acronym	13
acronymlists	15
acronyms	14
automake	24
compatible-2.07	25
compatible-3.07	25
counter	15
counter	15
description	218, 219
description	22
dua	216, 218, 219
dua	23
entrycounter	179, 180
true	9
entrycounter	9
entrycounterwithin	9
footnote 106–111, 214, 216, 218, 220	
footnote	22
hyperfirst	
false	106–111
hyperfirst	22
index	26
indexonlyfirst	364
indexonlyfirst	22
makeindex	144, 234
makeindex	23
nogroupskip	9
nohypertypes	16
nolist	228
nolist	8
nolong	228, 252
nolong	8
nomain	12, 13

nomain	13
nonumberlist	7
nonumberlist	7
nopostdot	9
noredefwarn	16
nostyles	8
nosuper	228
nosuper	8
notranslate	21
notree	228
notree	8
nowarn	16
numberedsection	6
numberline	5
numberline	5
numbers	26
order	23
sanitize	19, 58, 133
sanitize	21
sanitizesort	20
savenumberlist	7
savewrites	25, 361, 362
false	148
true	150, 156
savewrites	25
section	6, 37
section	6
seeautonumberlist	7
shotcuts	23
smallcaps	23
smaller	23
sort	
def	10
standard	10
use	10
sort	10
style	7, 228
style	7
subentrycounter	179, 181
subentrycounter	9
symbols	25
toc	5
true	5
toc	5
translate	21
false	21
translate	21
translator	21
ucmark	9

xindy	24, 144, 234	savewrites (option)	25
xindy	24	sc-short-long (acrstyle)	202
xindygloss	24	sc-short-long-desc (acrstyle) ..	
xindynoglsnumbers	24	203, 331
\pagelistname	30	\scantokens	49
parent (key)	60	\section	49
\PGLS	239	section (option)	6
\Pgls	238	see (key)	60
\pgls	236	seeautonumberlist (option)	7
\PGLSpl	240	\seename	30
\Pglspl	238	\SetAcronymLists	15
\pglspl	237	\SetAcronymStyle	14, 225
\phantomsection	36, 37	\setacronymstyle	199
plural (key)	58	\SetCustomDisplayStyle	226
pluralaccess (key)	309	\SetCustomStyle	227
polyglossia package	28, 30	\SetDefaultAcronymDisplayStyle	
\printacronyms	13	210
\PrintChanges	5	\SetDefaultAcronymStyle	211
\printglossaries	12,	\SetDescriptionAcronymDisplayStyle	
35, 53, 56, 57, 152, 163, 164, 244		216
\printglossary	35,	\SetDescriptionAcronymStyle	218
54, 57, 152, 163, 164, 166, 177, 244		\SetDescriptionDUAAcronymDisplayStyle	
\printglossary options		215
entrycounter	178	\SetDescriptionDUAAcronymStyle	
nogroupskip	178	216
nonumberlist	179	\SetDescriptionFootnoteAcronymDisplayStyle	
nopostdot	178	212
numberedsection	177	\SetDescriptionFootnoteAcronymStyle	
style	177	214
subentrycounter	179	\SetDUADisplayStyle	224
title	177	\SetDUASyle	225
toctitle	177	\setentrycounter	188
type	12, 163, 177	\SetFootnoteAcronymDisplayStyle	
\printnoidxglossaries	164	219
\printnoidxglossary	164, 177	\SetFootnoteAcronymStyle ...	220
\printnoidxglossary options		\SetGenericNewAcronym	197
sort	179	\setglossarypreamble	35
\provideglossaryentry	64	\setglossarysection	37
		\setglossarystyle	188
		\setglossentrycompatibility	185
		\SetSmallAcronymDisplayStyle	221
		\SetSmallAcronymStyle	223
		\setStyleFile	32, 33
		\setupglossaries	27
		short (key)	61
		short-long (acrstyle)	200, 329
		short-long-desc (acrstyle) .	203, 331
		shortaccess (key)	309
		shortplural (key)	61

R

\renewacronymstyle	199
\renewglossarystyle	189
\roman	43

S

\s@glshyp@opt	92
\s@newglossary	54
sanitize (option)	21
sanitizesort (option)	20
savenumberlist (option)	7

shortpluralaccess (key)	309	sm-short-long-desc (acrstyle) ..	
shotcuts (option)	23	203, 331
\showacronymlists	232	smallcaps (option)	23
\showglocounter	229	smaller (option)	23
\showglodesc	231	\SmallNewAcronymDef	222, 341
\showglodescaccess	343	sort (key)	58
\showglodescplural	231	sort (option)	10
\showglodescpluralaccess ...	343	style (option)	7
\showglofirst	229	subentrycounter (option)	9
\showglofirstaccess	343	\subglossentry	183
\showglofirsttpl	229	\subitem	280
\showglofirstpluralaccess ..	343	sublistdotted (style)	251
\showgloflag	232	\subsubitem	280
\showgloindex	232	super (style)	267
\showglolevel	228	super3col (style)	269
\showgloloclist	232	super3colborder (style)	270
\showglolong	232	super3colheader (style)	270
\showglolongaccess	344	super3colheaderborder (style) .	270
\showglolongpluralaccess ...	344	super4col (style)	271
\showglongame	231	super4colborder (style)	272
\showglongameaccess	342	super4colheader (style)	271
\showgloparent	228	super4colheaderborder (style) .	272
\showgloplural	229	superborder (style)	268
\showglopluralaccess	343	superheader (style)	268
\showgloshort	232	superheaderborder (style)	269
\showgloshortaccess	343	superragged (style)	274
\showgloshortpluralaccess ..	343	superragged3col (style)	276
\showglosort	231	superragged3colborder (style) .	277
\showglossaries	233	superragged3colheader (style) .	277
\showglossarycounter	233	superraggedborder (style)	275
\showglossaryentries	233	superraggedheader (style)	275
\showglossaryin	233	superraggedheaderborder (style)	276
\showglossaryout	233	superraggedright3colheaderborder	
\showglossarytitle	233	(style)	277
\showglosymbol	231	supertabular (environment) ...	
\showglosymbolaccess	343	8, 228, 267–279
\showglosymbolplural	231	supertabular package ..	8, 228, 267, 274
\showglosymbolpluralaccess .	343	symbol (key)	59
\showglotext	229	symbolaccess (key)	309
\showglotextaccess	343	\symbolname	30
\showglotype	229	symbolplural (key)	59
\showglouserii	230	symbolpluralaccess (key)	309
\showglouseriii	230	symbols (option)	25
\showglouseriv	230		
\showglouserv	230		
\showglouservi	230		
sm-short-long (acrstyle)	202		

T

text (key)	58
textaccess (key)	308
textcase package	3
\theequation	161

theglossary (environment) ... 5,
 16, 35, 183, 189, 265, 266, 282–285
 \theHequation 161
 theindex (environment) 280
 toc (option) 5
 \translate 30
 translate (option) 21
 translator package 12–14,
 26, 28, 30, 31, 164, 344, 353–356
 tree (style) 281
 treegroup (style) 282
 treehypergroup (style) 282
 treenoname (style) 283
 treenonamegroup (style) 284
 treenonamehypergroup (style) .. 284
 type (key) 59

U

ucmark (option) 9
 user1 (key) 60
 user2 (key) 60

user3 (key) 61
 user4 (key) 61
 user5 (key) 61
 user6 (key) 61

W

\warn@nomakeglossaries 150
 \warn@noprintglossary 163
 \writeist 33, 40, 41, 44, 142, 288, 290

X

\xcapitalisewords 243
 \xglsaccsupp 313
 xindy 357
 xindy 10, 23, 24, 33,
 39, 42, 44, 46, 47, 77, 103, 141,
 142, 144, 160, 166, 186, 234, 289
 xindy (option) 24
 xindygloss (option) 24
 xindynoglsnumbers (option) 24
 \xmakefirsttuc 242
 xspace package 4, 192