

Documented Code For glossaries v4.10

Nicola L.C. Talbot

Dickimaw Books

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

2014-08-27

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 v4.10: \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	140
1.12 Creating associated files	142
1.13 Writing information to associated files	157
1.14 Glossary Entry Cross-References	162
1.15 Displaying the glossary	164
1.16 Acronyms	193
1.17 Predefined acronym styles	198
1.18 Predefined Glossary Styles	229
1.19 Debugging Commands	229
1.20 Compatibility with version 2.07 and below	235
2 Prefix Support (glossaries-prefix Code)	235
3 Mfirstuc Documented Code	241
4 Mfirstuc-english Documented Code	244
5 Glossary Styles	245
5.1 Glossary hyper-navigation definitions (glossary-hypernav package)	245
5.2 In-line Style (glossary-inline.sty)	247
5.3 List Style (glossary-list.sty)	249
5.4 Glossary Styles using longtable (the glossary-long package)	252
5.5 Glossary Styles using longtable (the glossary-longragged package)	259
5.6 Glossary Styles using multicol (glossary-mcols.sty)	264
5.7 Glossary Styles using supertabular environment (glossary-super package)	268
5.8 Glossary Styles using supertabular environment (glossary-superragged package)	275
5.9 Tree Styles (glossary-tree.sty)	281

6 glossaries-compatible-207	288
7 Accessibility Support (glossaries-accsupp Code)	308
7.1 Defining Replacement Text	309
7.2 Accessing Replacement Text	313
7.3 Displaying the Glossary	328
7.4 Acronyms	329
7.5 Debugging Commands	343
8 Multi-Lingual Support	345
8.1 Babel Captions	345
8.2 Polyglossia Captions	351
8.3 Brazilian Dictionary	354
8.4 Danish Dictionary	354
8.5 Dutch Dictionary	355
8.6 English Dictionary	355
8.7 French Dictionary	355
8.8 German Dictionary	355
8.9 Irish Dictionary	356
8.10 Italian Dictionary	356
8.11 Magyar Dictionary	356
8.12 Polish Dictionary	357
8.13 Serbian Dictionary	357
8.14 Spanish Dictionary	357
Glossary	357
Change History	358
Index	381

1 Main Package Code

1.1 Package Definition

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

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries}[2014/08/27 v4.10 (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`. (As from v4.10, this isn't actually needed any more, but it's kept for backward compatibility.)

```

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

```

This code is only required for xindy:

```

1119 \ifglxindy

```

`\@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 }%

```

@appendix-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 `\@this@glo@`.

```

1309 \newcommand*\forallglsentries[3][\@glo@types]{%
1310   \expandafter\forallglossaries\expandafter[#1]{\@this@glo@}%
1311   {%
1312     \forallsentries[\@this@glo@]{#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\for glsentries\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][\gldefaulttype]{%
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}}
```

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

```
1650 \newcommand*\@glsnodedesc{%
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{}
```

`\@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   }
```

```

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}{%\glspercentchar^^J%
1787     \expandafter\@gobble\string\{\glspercentchar^^J%
1788     \string\gls@defglossaryentry{\glsdetoklabel{#1}}\glspercentchar^^J%
1789     \expandafter\@gobble\string\{\glspercentchar%
1790   }%

```

Write key value information:

```

1791 \@for\@gls@map:=\@gls@keymap\do
1792 {%
1793   \edef\glo@value{\expandafter\expandonce
1794     \csname glo@\glsdetoklabel{#1}\@expandafter
1795     \@secondoftwo\@gls@map\endcsname}%
1796   \@onelevel@sanitize\glo@value
1797   \immediate\write\@gls@deffile

```

```

1798    {%
1799        \expandafter\@firstoftwo\@gls@map
1800        =\expandafter\@gobble\string\{\gls@value\expandafter\@gobble\string\},%
1801        \glspercentchar%
1802    }%
1803 }%

```

Provide hook:

```

1804 \glswritedefhook
1805 \immediate\write\@gls@deffile
1806 {%
1807     \glspercentchar^^J%
1808     \expandafter\@gobble\string\}\glspercentchar^^J%
1809     \expandafter\@gobble\string\}\glspercentchar%
1810 }%
1811 }

```

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

```

1812 \newcommand*{\@gls@keymap}{%
1813     {name}{name},%
1814     {sort}{sortvalue},% unescaped sort value
1815     {type}{type},%
1816     {first}{first},%
1817     {firstplural}{firstpl},%
1818     {text}{text},%
1819     {plural}{plural},%
1820     {description}{desc},%
1821     {descriptionplural}{descplural},%
1822     {symbol}{symbol},%
1823     {symbolplural}{symbolplural},%
1824     {user1}{useri},%
1825     {user2}{userii},%
1826     {user3}{useriii},%
1827     {user4}{useriv},%
1828     {user5}{userv},%
1829     {user6}{uservi},%
1830     {long}{long},%
1831     {longplural}{longpl},%
1832     {short}{short},%
1833     {shortplural}{shortpl},%
1834     {counter}{counter},%
1835     {parent}{parent}%
1836 }

```

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

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

```
1837 \newcommand*{\@gls@fetchfield}[2]{%
```

Ensure user field name is fully expanded

```
1838 \edef\@gls@thisval{#2}%
```

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

```
1839 \@for\@gls@map:=\@gls@keymap\do{%
```

```
1840 \edef\@this@key{\expandafter\@firstoftwo\@gls@map}%
```

```
1841 \ifdefequal{\@this@key}{\@gls@thisval}%
```

```
1842 {%
```

Found it.

```
1843 \edef#1{\expandafter\@secondoftwo\@gls@map}%
```

Break out of loop.

```
1844 \@endfortrue
```

```
1845 }%
```

```
1846 {}%
```

```
1847 }%
```

```
1848 }
```

```
\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.

```
1849 \newcommand*{\glsaddkey}{\@ifstar\@sglsaddkey\@glsaddkey}
```

Starred version switches on expansion for this key.

```
1850 \newcommand*{\@sglsaddkey}[1]{%
```

```
1851 \key@ifundefined{glossentry}{#1}%
```

```
1852 {%
```

```
1853 \expandafter\newcommand\expandafter*\expandafter
```

```
1854 {\csname gls@assign@#1@field\endcsname}[2]{%
```

```
1855 \@gls@expand@field{##1}{#1}{##2}%
```

```
1856 }%
```

```
1857 }%
```

```
1858 {}%
```

```
1859 \@glsaddkey{#1}%
```

```
1860 }
```

Unstarred version doesn't override default expansion.

```
1861 \newcommand*{\@glsaddkey}[7]{%
```

Check the specified key doesn't already exist.

```
1862 \key@ifundefined{glossentry}{#1}%
```

```
1863 {%
```

Set up the key.

```
1864 \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
```

```
1865 \appto\@gls@keymap{, #1}{#1}}%
```

Set the default value.

```
1866 \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
```

Assignment code.

```
1867 \appto\@newglossaryentryposthook{%
1868 \letcs{\@glo@tmp}{@glo@#1}%
1869 \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
1870 }%
```

Define the no-link commands.

```
1871 \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
1872 \newcommand*{#4}[1]{\@Gls@entry@field{##1}{#1}}%
```

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

```
1873 \ifcsdef{@gls@user@#1@}%
1874 {%
1875 \PackageError{glossaries}%
1876 {Can't define '\string#5' as helper command
1877 '\expandafter\string\csname @gls@user@#1@\endcsname' already exists}%
1878 }%
1879 }%
1880 {%
1881 \expandafter\newcommand\expandafter*\expandafter
1882 {\csname @gls@user@#1\endcsname}[2][]{%
1883 \new@ifnextchar[%
1884 {\csuse{@gls@user@#1@}{##1}{##2}}%
1885 {\csuse{@gls@user@#1@}{##1}{##2}[]}}%
1886 \csdef{@gls@user@#1@}##1##2[##3]{%
1887 \@gls@field@link{##1}{##2}{#3{##2}##3}%
1888 }%
1889 \newrobustcmd*{#5}{%
1890 \expandafter\@gls@hyp@opt\csname @gls@user@#1\endcsname}%
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 \expandafter\newcommand\expandafter*\expandafter
1901 {\csname @Gls@user@#1\endcsname}[2][]{%
1902 \new@ifnextchar[%
1903 {\csuse{@Gls@user@#1@}{##1}{##2}}%
1904 {\csuse{@Gls@user@#1@}{##1}{##2}[]}}%
1905 \csdef{@Gls@user@#1@}##1##2[##3]{%
```

```

1906      \@gls@field@link{##1}{##2}{#4{##2}##3}%
1907      }%
1908      \newrobustcmd*{#6}{%
1909      \expandafter\@gls@hyp@opt\csname @Gls@user@#1\endcsname}%
1910      }%

```

Finally the all caps version:

```

1911      \ifcsdef{@GLS@user@#1@}%
1912      {%
1913      \PackageError{glossaries}%
1914      {Can't define '\string#7' as helper command
1915      '\expandafter\string\csname @GLS@user@#1@\endcsname' already exists}%
1916      }%
1917      }%
1918      {%
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      \newrobustcmd*{#7}{%
1928      \expandafter\@gls@hyp@opt\csname @GLS@user@#1\endcsname}%
1929      }%
1930      }%
1931      {%
1932      \PackageError{glossaries}{Key '#1' already exists}{}%
1933      }%
1934      }

```

\gls@writedefhook

```

1935 \newcommand*{\gls@writedefhook}{}

```

\gls@assign@desc

```

1936 \newcommand*{\gls@assign@desc}[1]{%
1937   \gls@assign@field{#1}{desc}{\@glo@desc}%
1938   \gls@assign@field{\@glo@desc}{#1}{descplural}{\@glo@descplural}%
1939 }

```

ongnewglossaryentry

```

1940 \newcommand{\longnewglossaryentry}[3]{%
1941   \glsdoifnoexists{#1}%
1942   {%
1943     \bgroup
1944     \let\@org@newglossaryentryprehook\@newglossaryentryprehook
1945     \long\def\@newglossaryentryprehook{%

```

```

1946      \long\def\@glo@desc{#3\leavevmode\unskip\nopostdesc}%
1947      \@org@newglossaryentryprehook
1948  }%
1949      \renewcommand*\@gls@assign@desc}[1]{%
1950          \global\cslet{glo@glstetoklabel{#1}@desc}{\@glo@desc}%
1951          \global\cslet{glo@glstetoklabel{#1}@descplural}{\@glo@desc}%
1952      }
1953      \gls@defglossaryentry{#1}{#2}%
1954  \egroup
1955 }
1956 }

```

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

```
1957 \@onlypreamble{\longnewglossaryentry}
```

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

```

1958 \newcommand{\longprovideglossaryentry}[3]{%
1959   \ifglstryexists{#1}{}%
1960   {\longnewglossaryentry{#1}{#2}{#3}}%
1961 }
1962 \@onlypreamble{\longprovideglossaryentry}

```

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

Defines a new entry without checking if it already exists.

```
1963 \newcommand{\gls@defglossaryentry}[2]{%
```

Store label

```
1964   \edef\@glo@label{\glstetoklabel{#1}}%
```

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

```
1965   \let\glslabel\@glo@label
```

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

```
1966   \let\@glo@name\@glsnname
```

```
1967   \let\@glo@desc\@glsnodesc
```

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

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

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

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

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

```
1973   \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.)

```
1974 \let\@glo@first\@gls@default@value
1975 \let\@glo@firstplural\@gls@default@value
```

Set the default sort:

```
1976 \let\@glo@sort\@gls@default@value
```

Set the default counter:

```
1977 \let\@glo@counter\@gls@default@value
```

```
1978 \def\@glo@see{}%
```

```
1979 \def\@glo@parent{}%
```

```
1980 \def\@glo@prefix{}%
```

```
1981 \def\@glo@useri{}%
```

```
1982 \def\@glo@userii{}%
```

```
1983 \def\@glo@useriii{}%
```

```
1984 \def\@glo@useriv{}%
```

```
1985 \def\@glo@userv{}%
```

```
1986 \def\@glo@uservi{}%
```

```
1987 \def\@glo@short{}%
```

```
1988 \def\@glo@shortpl{}%
```

```
1989 \def\@glo@long{}%
```

```
1990 \def\@glo@longpl{}%
```

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

```
1991 \@newglossaryentryprehook
```

Extract key-val information from third parameter:

```
1992 \setkeys{glossentry}{#2}%
```

Check there is a default glossary.

```
1993 \ifundef\glsdefaulttype
```

```
1994 {%
```

```
1995 \PackageError{glossaries}%
```

```
1996 {No default glossary type (have you used ‘nomain’?)}%
```

```
1997 {If you use package option ‘nomain’ you must define
```

```
1998 a new glossary before you can define entries}%
```

```
1999 }%
```

```
2000 {}%
```

Assign type. This must be fully expandable

```
2001 \gls@assign@field{\glsdefaulttype}{\@glo@label}{type}{\@glo@type}%
```

```
2002 \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.

```

2003 \ifcsundef{glolist@\glo@type}%
2004 {%
2005     \PackageError{glossaries}%
2006     {Glossary type ‘\@glo@type’ has not been defined}%
2007     {You need to define a new glossary type, before making entries
2008      in it}%
2009 }%
2010 {%

```

Check if it's an ignored glossary

```

2011 \ifignoredglossary\@glo@type
2012 {%

```

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

```

2013 \ifx\@glo@desc\glsnodel
2014 \let\@glo@desc\empty
2015 \fi
2016 }%
2017 {%
2018 }%
2019 \protected@edef\@glolist@{\csname glolist@\@glo@type\endcsname}%
2020 \expandafter\xdef\csname glolist@\@glo@type\endcsname{%
2021 \@glolist@{\@glo@label},}%
2022 }%

```

Initialise level to 0.

```

2023 \gls@level=0\relax

```

Has this entry been assigned a parent?

```

2024 \ifx\@glo@parent\empty

```

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

```

2025 \expandafter\gdef\csname glo@\@glo@label @parent\endcsname{}%
2026 \else

```

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

```

2027 \ifdefequal\@glo@label\@glo@parent%
2028 {%
2029     \PackageError{glossaries}{Entry ‘\@glo@label’ can't be its own parent}{}%
2030     \def\@glo@parent{}%
2031     \expandafter\gdef\csname glo@\@glo@label @parent\endcsname{}%
2032 }%
2033 {%

```

Check the parent exists:

```

2034 \ifglentryexists{\@glo@parent}%
2035 {%

```

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

```

2036 \expandafter\xdef\csname glo@\@glo@label @parent\endcsname{%
2037 \@glo@parent}%

```


Determine level.

```
2038      \gls@level=\csname glo@\@glo@parent @level\endcsname\relax
2039      \advance\gls@level by 1\relax
```

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

```
2040      \ifx\@glo@name\@glsnoname
2041      \expandafter\let\expandafter\@glo@name
2042      \csname glo@\@glo@parent @name\endcsname
```

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

```
2043      \ifx\@glo@plural\@gls@default@value
2044      \expandafter\let\expandafter\@glo@plural
2045      \csname glo@\@glo@parent @plural\endcsname
2046      \fi
2047      \fi
2048      }%
2049      {%
```

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

```
2050      \PackageError{glossaries}%
2051      {%
2052      Invalid parent '\@glo@parent'
2053      for entry '\@glo@label' - parent doesn't exist%
2054      }%
2055      {%
2056      Parent entries must be defined before their children%
2057      }%
2058      \def\@glo@parent{%
2059      \expandafter\gdef\csname glo@\@glo@label @parent\endcsname{%
2060      }%
2061      }%
2062      \fi
```

Set the level for this entry

```
2063      \expandafter\xdef\csname glo@\@glo@label @level\endcsname{\number\gls@level}%
```

Define commands associated with this entry:

```
2064      \gls@assign@field{\@glo@name}{\@glo@label}{sortvalue}{\@glo@sort}%
2065      \letcs\@glo@sort{glo@\@glo@label @sortvalue}%
2066      \gls@assign@field{\@glo@name}{\@glo@label}{text}{\@glo@text}%
2067      \expandafter\gls@assign@field\expandafter
2068      {\csname glo@\@glo@label @text\endcsname\glspluralsuffix}%
2069      {\@glo@label}{plural}{\@glo@plural}%
2070      \expandafter\gls@assign@field\expandafter
2071      {\csname glo@\@glo@label @text\endcsname}%
2072      {\@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.

```
2073      \ifx\@glo@first\@gls@default@value
```

```

2074 \expandafter\gls@assign@field\expandafter
2075 {\csname glo@\@glo@label @plural\endcsname}%
2076 {\@glo@label}{firstpl}{\@glo@firstplural}%
2077 \else
2078 \expandafter\gls@assign@field\expandafter
2079 {\csname glo@\@glo@label @first\endcsname\glspluralsuffix}%
2080 {\@glo@label}{firstpl}{\@glo@firstplural}%
2081 \fi

2082 \ifcsundef{@glo@type@\@glo@type @counter}%
2083 {%
2084 \def\@glo@defaultcounter{\glscounter}%
2085 }%
2086 {%
2087 \letcs\@glo@defaultcounter{@glo@type@\@glo@type @counter}%
2088 }%
2089 \gls@assign@field{\@glo@defaultcounter}{\@glo@label}{counter}{\@glo@counter}%
2090 \gls@assign@field{}{\@glo@label}{useri}{\@glo@useri}%
2091 \gls@assign@field{}{\@glo@label}{userii}{\@glo@userii}%
2092 \gls@assign@field{}{\@glo@label}{useriii}{\@glo@useriii}%
2093 \gls@assign@field{}{\@glo@label}{useriv}{\@glo@useriv}%
2094 \gls@assign@field{}{\@glo@label}{userv}{\@glo@userv}%
2095 \gls@assign@field{}{\@glo@label}{uservi}{\@glo@uservi}%
2096 \gls@assign@field{}{\@glo@label}{short}{\@glo@short}%
2097 \gls@assign@field{}{\@glo@label}{shortpl}{\@glo@shortpl}%
2098 \gls@assign@field{}{\@glo@label}{long}{\@glo@long}%
2099 \gls@assign@field{}{\@glo@label}{longpl}{\@glo@longpl}%
2100 \ifx\@glo@name\@glsnoname
2101 \@glsnoname
2102 \let\@glo@name\@gls@default@value
2103 \fi
2104 \gls@assign@field{}{\@glo@label}{name}{\@glo@name}%

```

Set default numberlist if not defined:

```

2105 \ifcsundef{glo@\@glo@label @numberlist}%
2106 {%
2107 \csxdef{glo@\@glo@label @numberlist}{%
2108 \noexpand\@gls@missingnumberlist{\@glo@label}}%
2109 }%
2110 {}%

```

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.

```

2111 \def\@glo@@desc{\@glo@first}%
2112 \ifx\@glo@desc\@glo@@desc
2113 \let\@glo@desc\@glo@first
2114 \fi
2115 \ifx\@glo@desc\@glsnodesc
2116 \@glsnodesc

```

```

2117 \let\@glodesc\@gls@default@value
2118 \fi
2119 \gls@assign@desc{\@glo@label}%

```

Set the sort key for this entry:

```

2120 \@gls@defsort{\@glo@type}{\@glo@label}%

2121 \def\@glo@@symbol{\@glo@text}%
2122 \ifx\@glo@symbol\@glo@@symbol
2123 \let\@glo@symbol\@glo@text
2124 \fi
2125 \gls@assign@field{\relax}{\@glo@label}{symbol}{\@glo@symbol}%
2126 \expandafter
2127 \gls@assign@field\expandafter
2128 {\csname glo@\@glo@label @symbol\endcsname}
2129 {\@glo@label}{symbolplural}{\@glo@symbolplural}%

```

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

```

2130 \expandafter\xdef\csname glo@\@glo@label @flagfalse\endcsname{%
2131 \noexpand\global
2132 \noexpand\let\expandafter\noexpand
2133 \csname ifglo@\@glo@label @flag\endcsname\noexpand\iffalse
2134 }%
2135 \expandafter\xdef\csname glo@\@glo@label @flagtrue\endcsname{%
2136 \noexpand\global
2137 \noexpand\let\expandafter\noexpand
2138 \csname ifglo@\@glo@label @flag\endcsname\noexpand\iftrue
2139 }%
2140 \csname glo@\@glo@label @flagfalse\endcsname

```

Sort out any cross-referencing if required.

```

2141 \ifdefvoid\@glo@see
2142 {}%
2143 {%
2144 \protected@edef\@do@glsee{%
2145 \noexpand\@gls@fixbraces\noexpand\@glo@list\@glo@see
2146 \noexpand\@nil
2147 \noexpand\expandafter\noexpand\@glsee\noexpand\@glo@list{\@glo@label}}%
2148 \@do@glsee
2149 }%

```

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

```

2150 \ifignoredglossary\@glo@type
2151 {%
2152 \csdef{glo@\@glo@label @index}{}%
2153 }
2154 {%
2155 \do@glo@storeentry{\@glo@label}%
2156 }%

```

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

```
2157 \newglossaryentryposthook
2158 }
```

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

```
2159 \newcommand*{\@newglossaryentryprehook}{}%
```

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

```
2160 \newcommand*{\@newglossaryentryposthook}{}%
```

`\glsmoveentry` Moves entry whose label is given by first argument to the glossary named in the second argument.

```
2161 \newcommand*{\glsmoveentry}[2]{%
2162   \edef\@glo@thislabel{\glsetoklabel{#1}}%
2163   \edef\@glo@type{\csname glo@\@glo@thislabel @type\endcsname}%
2164   \def\@glo@list{,}%
2165   \for@gl@sentries[\@glo@type]{\@glo@label}%
2166   {%
2167     \ifdefequal\@glo@thislabel\@glo@label
2168     {}{\eappto\@glo@list{\@glo@label,}}%
2169   }%
2170   \cslet\@glo@list\@glo@type{\@glo@list}%
2171   \csdef\@glo@\@glo@thislabel @type{#2}%
2172 }
```

`\@glossaryentryfield` Indicate what command should be used to display each entry in the glossary. (This enables the glossaries-accsupp package to use `\accsuppglossaryentryfield` instead.)

```
2173 \ifglxindy
2174   \newcommand*{\@glossaryentryfield}{\string\glossentry}
2175 \else
2176   \newcommand*{\@glossaryentryfield}{\string\glossentry}
2177 \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.)

```
2178 \ifglxindy
2179   \newcommand*{\@glossarysubentryfield}{%
2180     \string\subglossentry}
2181 \else
2182   \newcommand*{\@glossarysubentryfield}{%
2183     \string\subglossentry}
2184 \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.)

```
2185 \newcommand{\@glo@storeentry}[1]{%
```

Escape makeindex/xindy special characters in the label:

```
2186 \edef\@glo@esclabel{#1}%
2187 \@gls@checkmkidxchars\@glo@esclabel
```

Get the sort string and escape any special characters

```
2188 \protected@edef\@glo@sort{\csname glo@#1@sort\endcsname}%
2189 \@gls@checkmkidxchars\@glo@sort
```

Same again for the name string. Escape any special characters in the prefix

```
2190 \@gls@checkmkidxchars\@glo@prefix
```

Get the parent, if one exists

```
2191 \edef\@glo@parent{\csname glo@#1@parent\endcsname}%
```

Write the information to the glossary file.

```
2192 \ifglxindy
```

Store using xindy syntax.

```
2193 \ifx\@glo@parent\@empty
```

Entry doesn't have a parent

```
2194 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
2195 (\string"\@glo@sort\string" %
2196 \string"\@glo@prefix\@glossaryentryfield{\@glo@esclabel}\string") %
2197 }%
2198 \else
```

Entry has a parent

```
2199 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
2200 \csname glo@\@glo@parent @index\endcsname
2201 (\string"\@glo@sort\string" %
2202 \string"\@glo@prefix\@glossarysubentryfield
2203 {\csname glo@#1@level\endcsname}{\@glo@esclabel}\string") %
2204 }%
2205 \fi
2206 \else
```

Store using makeindex syntax.

```
2207 \ifx\@glo@parent\@empty
```

Sanitize \@glo@prefix

```
2208 \@onelevel@sanitize\@glo@prefix
```

Entry doesn't have a parent

```
2209 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
2210 \@glo@sort\@gls@actualchar\@glo@prefix
2211 \@glossaryentryfield{\@glo@esclabel}%
```

```

2212     }%
2213   \else
      Entry has a parent
2214     \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
2215       \csname glo@\@glo@parent @index\endcsname\@gls@levelchar
2216       \@glo@sort\@gls@actualchar\@glo@prefix
2217       \@glossarysubentryfield
2218       {\csname glo@#1@level\endcsname}\@glo@esclabel}%
2219     }%
2220   \fi
2221 \fi
2222 }

```

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`

```

2223 \AtBeginDocument{%
2224   \ifpackageloaded{amsmath}%
2225   {\let\gls@ifnotmeasuring\@gls@ifnotmeasuring}%
2226   {}}%
2227 }
2228 \newcommand*{\@gls@ifnotmeasuring}[1]{%
2229   \ifmeasuring@
2230   \else
2231     #1%
2232   \fi
2233 }
2234 \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.

```

2235 \newcommand*{\glsreset}[1]{%
2236   \gls@ifnotmeasuring
2237   {%
2238     \glsdoifexists{#1}%
2239     {%
2240       \expandafter\global\csname glo@\glsdetoklabel{#1}@flagfalse\endcsname
2241     }%
2242   }%
2243 }

```

`\glslocalreset` As above, but with only a local effect:

```
2244 \newcommand*{\glslocalreset}[1]{%
2245   \gls@ifnotmeasuring
2246   {%
2247     \glsdoifexists{#1}%
2248     {%
2249       \expandafter\let\csname ifglo@\glsdetoklabel{#1}@flag\endcsname\iffalse
2250     }%
2251   }%
2252 }
```

`\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.

```
2253 \newcommand*{\glsunset}[1]{%
2254   \gls@ifnotmeasuring
2255   {%
2256     \glsdoifexists{#1}%
2257     {%
2258       \expandafter\global\csname glo@\glsdetoklabel{#1}@flagtrue\endcsname
2259     }%
2260   }%
2261 }
```

`\glslocalunset` As above, but with only a local effect:

```
2262 \newcommand*{\glslocalunset}[1]{%
2263   \gls@ifnotmeasuring
2264   {%
2265     \glsdoifexists{#1}%
2266     {%
2267       \expandafter\let\csname ifglo@\glsdetoklabel{#1}@flag\endcsname\iftrue
2268     }%
2269   }%
2270 }
```

Reset all entries for the named glossaries (supplied in a comma-separated list).

Syntax: `\glsresetall[<glossary-list>]`

`\glsresetall`

```
2271 \newcommand*{\glsresetall}[1][\@glo@types]{%
2272   \forallglsentries[#1]{\@glsentry}%
2273   {%
2274     \glsreset{\@glsentry}%
2275   }%
2276 }
```

As above, but with only a local effect:

`\glslocalresetall`

```
2277 \newcommand*{\glslocalresetall}[1][\@glo@types]{%
```

```

2278 \forallglsentries[#1]{\@glsentry}%
2279 {%
2280     \glslocalreset{\@glsentry}%
2281 }%
2282 }

```

Unset all entries for the named glossaries (supplied in a comma-separated list).
 Syntax: `\glsunsetall[<glossary-list>]`

`\glsunsetall`

```

2283 \newcommand*{\glsunsetall}[1][\@gls@types]{%
2284     \forallglsentries[#1]{\@glsentry}%
2285     {%
2286         \glsunset{\@glsentry}%
2287     }%
2288 }

```

As above, but with only a local effect:

`\glslocalunsetall`

```

2289 \newcommand*{\glslocalunsetall}[1][\@gls@types]{%
2290     \forallglsentries[#1]{\@glsentry}%
2291     {%
2292         \glslocalunset{\@glsentry}%
2293     }%
2294 }

```

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`

```

2295 \newcommand*{\loadglsentries}[2][\@gls@default]{%
2296     \let\@gls@default\glsdefaulttype
2297     \def\glsdefaulttype{#1}\input{#2}%
2298     \let\glsdefaulttype\@gls@default
2299 }

```

¹and any other valid \LaTeX code that can be used in the preamble.

`\loadglsentries` can only be used in the preamble:
2300 `\@onlypreamble{\loadglsentries}`

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`

2301 `\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

2302 `\newcommand*{\glsentryfmt}{%`

2303 `\@@gls@default@entryfmt\glsdisplayfirst\glsdisplay`

2304 `}`

Format that provides backwards compatibility:

2305 `\newcommand*{\@@gls@default@entryfmt}[2]{%`

2306 `\ifdefempty\glscustomtext`

2307 `{%`

2308 `\glsifplural`

2309 `{%`

Plural form

2310 `\glscapscase`

2311 `{%`

Don’t adjust case

2312 `\ifglsused\glslabel`

2313 `{%`

Subsequent use

2314 `#2{\glsentryplural{\glslabel}}%`

2315 `{\glsentrydescplural{\glslabel}}%`

2316 `{\glsentrysymbolplural{\glslabel}}{\glsinsert}%`

2317 `}%`

2318 `{%`

First use

2319 `#1{\glsentryfirstplural{\glslabel}}%`

2320 `{\glsentrydescplural{\glslabel}}%`

2321 `{\glsentrysymbolplural{\glslabel}}{\glsinsert}%`

```

2322     }%
2323     }%
2324     {%

```

Make first letter upper case

```

2325     \ifglsused\glslabel
2326     {%

```

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 `\defglsentryfmt`, which avoids the issues caused by fragile commands.)

```

2327     \ifbool{glscompatible-3.07}%
2328     {%
2329     \protected@edef\@glo@etext{%
2330     #2{\glsentryplural{\glslabel}}%
2331     {\glsentrydescplural{\glslabel}}%
2332     {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2333     \xmakefirstuc\@glo@etext
2334     }%
2335     {%
2336     #2{\Glsentryplural{\glslabel}}%
2337     {\Glsentrydescplural{\glslabel}}%
2338     {\Glsentrysymbolplural{\glslabel}}{\glsinsert}%
2339     }%
2340     }%
2341     {%

```

First use

```

2342     \ifbool{glscompatible-3.07}%
2343     {%
2344     \protected@edef\@glo@etext{%
2345     #1{\glsentryfirstplural{\glslabel}}%
2346     {\glsentrydescplural{\glslabel}}%
2347     {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2348     \xmakefirstuc\@glo@etext
2349     }%
2350     {%
2351     #1{\Glsentryfirstplural{\glslabel}}%
2352     {\Glsentrydescplural{\glslabel}}%
2353     {\Glsentrysymbolplural{\glslabel}}{\glsinsert}%
2354     }%
2355     }%
2356     }%
2357     {%

```

Make all upper case

```

2358     \ifglsused\glslabel
2359     {%

```

Subsequent use

```

2360      \mfirstucMakeUppercase{#2{\glsentryplural{\glslabel}}}%
2361      {\glsentrydescplural{\glslabel}}}%
2362      {\glsentrysymbolplural{\glslabel}}{\glsinsert}}}%
2363  }%
2364  {%

```

First use

```

2365      \mfirstucMakeUppercase{#1{\glsentryfirstplural{\glslabel}}}%
2366      {\glsentrydescplural{\glslabel}}}%
2367      {\glsentrysymbolplural{\glslabel}}{\glsinsert}}}%
2368  }%
2369  }%
2370  }%
2371  {%

```

Singular form

```

2372      \glscapscase
2373      {%

```

Don't adjust case

```

2374      \ifglused\glslabel
2375      {%

```

Subsequent use

```

2376      #2{\glsentrytext{\glslabel}}}%
2377      {\glsentrydesc{\glslabel}}}%
2378      {\glsentrysymbol{\glslabel}}{\glsinsert}%
2379  }%
2380  {%

```

First use

```

2381      #1{\glsentryfirst{\glslabel}}}%
2382      {\glsentrydesc{\glslabel}}}%
2383      {\glsentrysymbol{\glslabel}}{\glsinsert}%
2384  }%
2385  }%
2386  {%

```

Make first letter upper case

```

2387      \ifglused\glslabel
2388      {%

```

Subsequent use

```

2389      \ifbool{glscompatible-3.07}%
2390      {%
2391      \protected@edef\@glo@etext{%
2392      #2{\glsentrytext{\glslabel}}}%
2393      {\glsentrydesc{\glslabel}}}%
2394      {\glsentrysymbol{\glslabel}}{\glsinsert}}}%
2395      \xmakefirstuc\@glo@etext
2396  }%
2397  {%

```

```

2398         #2{\Glsentrytext{\glslabel}}%
2399         {\glsentrydesc{\glslabel}}%
2400         {\glsentrysymbol{\glslabel}}{\glsinsert}%
2401     }%
2402 }%
2403 {%

```

First use

```

2404     \ifbool{glscompatible-3.07}%
2405     {%
2406         \protected@edef\@glo@etext{%
2407             #1{\glsentryfirst{\glslabel}}%
2408             {\glsentrydesc{\glslabel}}%
2409             {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2410         \xmakefirstuc\@glo@etext
2411     }%
2412 {%
2413     #1{\Glsentryfirst{\glslabel}}%
2414     {\glsentrydesc{\glslabel}}%
2415     {\glsentrysymbol{\glslabel}}{\glsinsert}%
2416 }%
2417 }%
2418 }%
2419 {%

```

Make all upper case

```

2420     \ifglsused\glslabel
2421     {%

```

Subsequent use

```

2422         \mfirstucMakeUppercase{#2{\glsentrytext{\glslabel}}%
2423         {\glsentrydesc{\glslabel}}%
2424         {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2425     }%
2426     {%

```

First use

```

2427         \mfirstucMakeUppercase{#1{\glsentryfirst{\glslabel}}%
2428         {\glsentrydesc{\glslabel}}%
2429         {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2430     }%
2431 }%
2432 }%
2433 }%
2434 {%

```

Custom text provided in \glsdisp

```

2435     \ifglsused{\glslabel}%
2436     {%

```

Subsequent use

```

2437     #2{\glscustomtext}%

```

```

2438      {\glsentrydesc{\glslabel}}}%
2439      {\glsentrysymbol{\glslabel}}{}}%
2440    }%
2441    {%

```

First use

```

2442      #1{\glscustomtext}%
2443      {\glsentrydesc{\glslabel}}}%
2444      {\glsentrysymbol{\glslabel}}{}}%
2445    }%
2446  }%
2447 }

```

`\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.

```

2448 \newcommand*{\glsgenentryfmt}{%
2449   \ifdefempty\glscustomtext
2450   {%
2451     \glsifplural
2452     {%

```

Plural form

```

2453     \glscapscase
2454     {%

```

Don't adjust case

```

2455     \ifglsused\glslabel
2456     {%

```

Subsequent use

```

2457     \glsentryplural{\glslabel}\glsinsert
2458     }%
2459     {%

```

First use

```

2460     \glsentryfirstplural{\glslabel}\glsinsert
2461     }%
2462     }%
2463     {%

```

Make first letter upper case

```

2464     \ifglsused\glslabel
2465     {%

```

Subsequent use.

```

2466     \Glsentryplural{\glslabel}\glsinsert
2467     }%
2468     {%

```

First use

```

2469     \Glsentryfirstplural{\glslabel}\glsinsert
2470     }%

```

```

2471     }%
2472     {%

    Make all upper case
2473     \ifglused\glslabel
2474     {%

    Subsequent use
2475     \mfirstucMakeUppercase
2476     {\glentryplural{\glslabel}\glsinsert}%
2477     }%
2478     {%

    First use
2479     \mfirstucMakeUppercase
2480     {\glentryfirstplural{\glslabel}\glsinsert}%
2481     }%
2482     }%
2483     }%
2484     {%

    Singular form
2485     \glscapscase
2486     {%

    Don't adjust case
2487     \ifglused\glslabel
2488     {%

    Subsequent use
2489     \glentrytext{\glslabel}\glsinsert
2490     }%
2491     {%

    First use
2492     \glentryfirst{\glslabel}\glsinsert
2493     }%
2494     }%
2495     {%

    Make first letter upper case
2496     \ifglused\glslabel
2497     {%

    Subsequent use
2498     \Glsentrytext{\glslabel}\glsinsert
2499     }%
2500     {%

    First use
2501     \Glsentryfirst{\glslabel}\glsinsert
2502     }%
2503     }%
2504     {%

```

Make all upper case

```
2505      \ifglsused\glslabel
2506      {%
```

Subsequent use

```
2507      \mfirstucMakeUppercase{\glsentrytext{\glslabel}\glsinsert}%
2508      }%
2509      {%
```

First use

```
2510      \mfirstucMakeUppercase{\glsentryfirst{\glslabel}\glsinsert}%
2511      }%
2512      }%
2513      }%
2514      }%
2515      {%
```

Custom text provided in \glsdisp. (The insert is most likely to be empty at this point.)

```
2516      \glscustomtext\glsinsert
2517      }%
2518 }
```

`\glsngenacfmt` Define a generic acronym format that uses the long and short keys (or their plurals) and `\acrfullformat`, `\firstacronymfont` and `\acronymfont`.

```
2519 \newcommand*{\glsngenacfmt}{%
2520   \ifdefempty\glscustomtext
2521   {%
2522     \ifglsused\glslabel
2523     {%
```

Subsequent use:

```
2524     \glsifplural
2525     {%
```

Subsequent plural form:

```
2526     \glscapscase
2527     {%
```

Subsequent plural form, don't adjust case:

```
2528     \acronymfont{\glsentryshortpl{\glslabel}}\glsinsert
2529     }%
2530     {%
```

Subsequent plural form, make first letter upper case:

```
2531     \acronymfont{\Glsentryshortpl{\glslabel}}\glsinsert
2532     }%
2533     {%
```

Subsequent plural form, all caps:

```
2534     \mfirstucMakeUppercase
2535     {\acronymfont{\glsentryshortpl{\glslabel}}\glsinsert}%
```

2536 }%
 2537 }%
 2538 {%

Subsequent singular form

2539 \glscapscase
 2540 {%

Subsequent singular form, don't adjust case:

2541 \acronymfont{\glstryshort{\glslabel}}\glsinsert
 2542 }%
 2543 {%

Subsequent singular form, make first letter upper case:

2544 \acronymfont{\Glsstryshort{\glslabel}}\glsinsert
 2545 }%
 2546 {%

Subsequent singular form, all caps:

2547 \mfirstucMakeUppercase
 2548 {\acronymfont{\glstryshort{\glslabel}}\glsinsert}%
 2549 }%
 2550 }%
 2551 }%
 2552 {%

First use:

2553 \glsifplural
 2554 {%

First use plural form:

2555 \glscapscase
 2556 {%

First use plural form, don't adjust case:

2557 \genplacrfullformat{\glslabel}{\glsinsert}%
 2558 }%
 2559 {%

First use plural form, make first letter upper case:

2560 \Genplacrfullformat{\glslabel}{\glsinsert}%
 2561 }%
 2562 {%

First use plural form, all caps:

2563 \mfirstucMakeUppercase
 2564 {\genplacrfullformat{\glslabel}{\glsinsert}}%
 2565 }%
 2566 }%
 2567 {%

First use singular form

2568 \glscapscase
 2569 {%

First use singular form, don't adjust case:

```
2570      \genacrfullformat{\glslabel}{\glsinsert}%
2571      }%
2572      {%
```

First use singular form, make first letter upper case:

```
2573      \Genacrfullformat{\glslabel}{\glsinsert}%
2574      }%
2575      {%
```

First use singular form, all caps:

```
2576      \mfirstucMakeUppercase
2577      {\genacrfullformat{\glslabel}{\glsinsert}}%
2578      }%
2579      }%
2580      }%
2581      }%
2582      {%
```

User supplied text.

```
2583      \glscustomtext
2584      }%
2585 }
```

`\genacrfullformat` `\genacrfullformat{<label>}{<insert>}`

The full format used by `\glsngenacfmt` (singular).

```
2586 \newcommand*{\genacrfullformat}[2]{%
2587   \glsentrylong{#1}#2\space
2588   (\protect\firstacronymfont{\glsentryshort{#1}})%
2589 }
```

`\Genacrfullformat` `\Genacrfullformat{<label>}{<insert>}`

As above but makes the first letter upper case.

```
2590 \newcommand*{\Genacrfullformat}[2]{%
2591   \protected@edef\gls@text{\genacrfullformat{#1}{#2}}%
2592   \xmakefirstuc\gls@text
2593 }
```

`\genplacrfullformat` `\genplacrfullformat{<label>}{<insert>}`

The full format used by `\glsngenacfmt` (plural).

```
2594 \newcommand*{\genplacrfullformat}[2]{%
2595   \glsentrylongpl{#1}#2\space
2596   (\protect\firstacronymfont{\glsentryshortpl{#1}})%
```

2597 }

`\Genplacrfullformat` `\Genplacrfullformat{<label>}{<insert>}`

As above but makes the first letter upper case.

```
2598 \newcommand*{\Genplacrfullformat}[2]{%
2599   \protected@edef\gls@text{\genplacrfullformat{#1}{#2}}%
2600   \xmakefirstuc\gls@text
2601 }
```

`\glsdisplayfirst` Deprecated. Kept for backward compatibility.

```
2602 \newcommand*{\glsdisplayfirst}[4]{#1#4}
```

`\glsdisplay` Deprecated. Kept for backward compatibility.

```
2603 \newcommand*{\glsdisplay}[4]{#1#4}
```

`\defglsdisplay` Deprecated. Kept for backward compatibility.

```
2604 \newcommand*{\defglsdisplay}[2][\glsdefaulttype]{%
2605   \GlossariesWarning{\string\defglsdisplay\space is now obsolete.^^J
2606   Use \string\defglsentryfmt\space instead}%
2607   \expandafter\def\csname gls@#1@display\endcsname##1##2##3##4{#2}%
2608   \edef\@gls@doentrydef{%
2609     \noexpand\defglsentryfmt[#1]{%
2610       \noexpand\ifcsdef{gls@#1@displayfirst}%
2611       {%
2612         \noexpand\@gls@default@entryfmt
2613         {\noexpand\csuse{gls@#1@displayfirst}}%
2614         {\noexpand\csuse{gls@#1@display}}%
2615       }%
2616       {%
2617         \noexpand\@gls@default@entryfmt
2618         {\noexpand\glsdisplayfirst}%
2619         {\noexpand\csuse{gls@#1@display}}%
2620       }%
2621     }%
2622   }%
2623   \@gls@doentrydef
2624 }
```

`\defglsdisplayfirst` Deprecated. Kept for backward compatibility.

```
2625 \newcommand*{\defglsdisplayfirst}[2][\glsdefaulttype]{%
2626   \GlossariesWarning{\string\defglsdisplayfirst\space is now obsolete.^^J
2627   Use \string\defglsentryfmt\space instead}%
2628   \expandafter\def\csname gls@#1@displayfirst\endcsname##1##2##3##4{#2}%
2629   \edef\@gls@doentrydef{%
2630     \noexpand\defglsentryfmt[#1]{%
2631       \noexpand\ifcsdef{gls@#1@display}%

```

```

2632      {%
2633      \noexpand\@@gls@default@entryfmt
2634      {\noexpand\csuse{gls@#1@displayfirst}}}%
2635      {\noexpand\csuse{gls@#1@display}}}%
2636      }%
2637      {%

2638      \noexpand\@@gls@default@entryfmt
2639      {\noexpand\csuse{gls@#1@displayfirst}}}%
2640      {\noexpand\glsdisplay}%
2641      }%
2642      }%
2643      }%
2644      \@gls@doentrydef
2645      }

```

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.

```

2646 \define@key{glslink}{counter}{%
2647   \ifcsundef{c@#1}%
2648   {%
2649     \PackageError{glossaries}%
2650     {There is no counter called ‘#1’}%
2651     {%
2652       The counter key should have the name of a valid counter
2653       as its value%
2654     }%
2655   }%
2656   {%
2657     \def\@gls@counter{#1}%
2658   }%
2659 }

```

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.

```
2660 \define@key{glslink}{format}{%
2661   \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.

```
2662 \define@boolkey{glslink}{hyper}[true]{}
```

Initialise hyper key.

```
2663 \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.

```
2664 \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.

```
2665 \newcommand*{\glslinkvar}[3]{#1}
```

\glsifhyper Now deprecated.

```
2666 \newcommand*{\glsifhyper}[2]{%
2667   \glslinkvar{#1}{#2}{#1}%
2668   \GlossariesWarning{\string\glsifhyper\space is deprecated. Did
2669     you mean \string\glsifhyperon\space or \string\glslinkvar?}%
2670 }
```

\@gls@hyp@opt Used by the commands such as \glslink to determine whether to modify the hyper option.

```
2671 \newcommand*{\@gls@hyp@opt}[1]{%
2672   \let\glslinkvar\@firstofthree
2673   \let\@gls@hyp@opt@cs#1\relax
2674   \ifstar{\s@gls@hyp@opt}%
2675   {\@ifnextchar+{\@firstoftwo{\p@gls@hyp@opt}}{#1}}%
2676 }
```

\s@gls@hyp@opt Starred version

```
2677 \newcommand*{\s@gls@hyp@opt}[1][ ]{%
```

```

2678 \let\glslinkvar\@secondofthree
2679 \@gls@hyp@opt@cs[hyper=false,#1]}

```

\p@gls@hyp@opt Plus version

```

2680 \newcommand*{\p@gls@hyp@opt}[1] [] {%
2681 \let\glslinkvar\@thirdofthree
2682 \@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

```

2683 \newrobustcmd*{\glslink}{%
2684 \@gls@hyp@opt\@gls@@link
2685 }

```

\@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.

```

2686 \newcommand*{\@gls@@link}[3] [] {%
2687 \ifglsentryexists{#2}%
2688 {%
2689 \let\do@gls@link@checkfirsthyper\relax
2690 \@gls@link[#1]{#2}{#3}%
2691 }{%
2692 \PackageError{glossaries}{Glossary entry ‘#2’ has not been
2693 defined}{You need to define a glossary entry before you
2694 can use it.}%

```

Display the specified text. (The entry doesn't exist so there's nothing to link it to.)

```

2695 \glstextformat{#3}%
2696 }%
2697 }

```

\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`.

```

2698 \newcommand*{\@gls@link@checkfirsthyper}{%
2699   \ifglsused{\glslabel}%
2700   {%
2701   }%
2702   {%
2703     \gls@checkisacronymlist\glstype
2704     \ifglshyperfirst
2705       \ifglsisacronymlist
2706         \ifglsacrfootnote
2707           \KV@glslink@hyperfalse
2708         \fi
2709       \fi
2710     \else
2711       \KV@glslink@hyperfalse
2712     \fi
2713   }%

```

Allow user to hook into this

```

2714   \glslinkcheckfirsthyperhook
2715 }

```

`checkfirsthyperhook` Allow used to hook into the `\gls@link@checkfirsthyper` macro

```

2716 \newcommand*{\glslinkcheckfirsthyperhook}{}

```

`\@gls@link`

```

2717 \def\@gls@link[#1]#2#3{%

```

Inserting `\leavevmode` suggested by Donald Arseneau (avoids problem with `tabularx`).

```

2718   \leavevmode
2719   \edef\glslabel{\glsdetoklabel{#2}}%

```

Save options in `\@gls@link@opts` and label in `\@gls@link@label`

```

2720   \def\@gls@link@opts{#1}%
2721   \let\@gls@link@label\glslabel
2722   \def\@glsnumberformat{\glsnumberformat}%
2723   \edef\@gls@counter{\csname glo@\glslabel @counter\endcsname}%

```

If this is in one of the “nohypertypes” glossaries, suppress the hyperlink by default

```

2724   \edef\glstype{\csname glo@\glslabel @type\endcsname}%

```

Save original setting

```

2725   \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper

```

Switch off hyper setting if the glossary type has been identified in `nohyperlist`.

```

2726   \expandafter\DTLifinlist\expandafter
2727     {\glstype}{\@gls@nohyperlist}%

```

```

2728    {%
2729        \KV@glslink@hyperfalse
2730    }%
2731    {%
2732    }%

```

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`.

```

2733    \do@glslink@checkfirsthyper
2734    \setkeys{glslink}{#1}%

    Define \glsifhyperon
2735    \ifKV@glslink@hyper
2736        \let\glsifhyperon\@firstoftwo
2737    \else
2738        \let\glsifhyperon\@secondoftwo
2739    \fi

```

Store the entry's counter in `\theglentrycounter`

```

2740    \@gls@saveentrycounter

```

Define sort key if necessary:

```

2741    \@gls@setsort{\glslabel}%

    (De-tok'ing done by \@do@wrglossary)
2742    \@do@wrglossary{#2}%
2743    \ifKV@glslink@hyper
2744        \@glslink{\glolinkprefix\glslabel}{\glstextformat{#3}}%
2745    \else

2746        \glstextformat{#3}%
2747    \fi

```

Restore original setting

```

2748    \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper
2749 }

```

`\glolinkprefix`

```

2750 \newcommand*{\glolinkprefix}{glo:}

```

`\glentrycounter` Set default value of entry counter

```

2751 \def\glentrycounter{\glscounter}%

```

`\@saveentrycounter` Need to check if using equation counter in align environment:

```

2752 \newcommand*{\@gls@saveentrycounter}{%
2753     \def\@gls@Hcounter{}%

    Are we using equation counter?
2754     \ifthenelse{\equal{\@gls@counter}{equation}}{%
2755     {

```

If we're in align environment, \xatlevel@ will be defined. (Can't test for \@currentenv as may be inside an inner environment.)

```

2756 \ifcsundef{xatlevel@}%
2757 {%
2758 \edef\theglsentrycounter{\expandafter\noexpand
2759 \csname the\@gls@counter\endcsname}%
2760 }%
2761 {%
2762 \ifx\xatlevel@\@empty
2763 \edef\theglsentrycounter{\expandafter\noexpand
2764 \csname the\@gls@counter\endcsname}%
2765 \else
2766 \savecounters@
2767 \advance\c@equation by 1\relax
2768 \edef\theglsentrycounter{\csname the\@gls@counter\endcsname}%

```

Check if hyperref version of this counter

```

2769 \ifcsundef{theH\@gls@counter}%
2770 {%
2771 \def\@gls@Hcounter{\theglsentrycounter}%
2772 }%
2773 {%
2774 \def\@gls@Hcounter{\csname theH\@gls@counter\endcsname}%
2775 }%
2776 \protected@edef\theHglentrycounter{\@gls@Hcounter}%
2777 \restorecounters@
2778 \fi
2779 }%
2780 }%
2781 {%

```

Not using equation counter so no special measures:

```

2782 \edef\theglsentrycounter{\expandafter\noexpand
2783 \csname the\@gls@counter\endcsname}%
2784 }%

```

Check if hyperref version of this counter

```

2785 \ifx\@gls@Hcounter\@empty
2786 \ifcsundef{theH\@gls@counter}%
2787 {%
2788 \def\theHglentrycounter{\theglsentrycounter}%
2789 }%
2790 {%
2791 \protected@edef\theHglentrycounter{\expandafter\noexpand
2792 \csname theH\@gls@counter\endcsname}%
2793 }%
2794 \fi
2795 }

```


`\@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.

```

2796 \def\@set@glo@numformat#1#2#3#4{%
2797   \expandafter\@glo@check@mkidxrangechar#3\@nil
2798   \protected@edef#1{%
2799     \@glo@prefix setentrycounter[#4]{#2}%
2800     \expandafter\string\csname\@glo@suffix\endcsname
2801   }%
2802   \@gls@checkmkidxchars#1%
2803 }

```

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.

```

2804 \def\@glo@check@mkidxrangechar#1#2\@nil{%
2805   \if#1(\relax
2806     \def\@glo@prefix{(%}
2807     \if\relax#2\relax
2808       \def\@glo@suffix{glsnumberformat}%
2809     \else
2810       \def\@glo@suffix{#2}%
2811     \fi
2812   \else
2813     \if#1)\relax
2814       \def\@glo@prefix{)%}
2815     \if\relax#2\relax
2816       \def\@glo@suffix{glsnumberformat}%
2817     \else
2818       \def\@glo@suffix{#2}%
2819     \fi
2820   \else
2821     \def\@glo@prefix{}\def\@glo@suffix{#1#2}%
2822   \fi
2823 }

```

`\@gls@escbsdq` Escape backslashes and double quote marks. The argument must be a control sequence.

```

2824 \newcommand*\@gls@escbsdq[1]{%
2825   \def\@gls@checkedmkidx{%}
2826   \let\gls@xdystring=#1\relax
2827   \@onelevel@sanitize\gls@xdystring
2828   \edef\do@gls@xdycheckbackslash{%
2829     \noexpand\@gls@xdycheckbackslash\gls@xdystring\noexpand\@nil
2830     \@backslashchar\@backslashchar\noexpand\null}%

```

```

2831 \do@gl@xdycheckbackslash
2832 \expandafter\@gl@updatechecked\@gl@checkedmkidx{\gl@xdystring}%
2833 \def\@gl@checkedmkidx{%
2834 \expandafter\@gl@xdycheckquote\gl@xdystring\@nil""\null
2835 \expandafter\@gl@updatechecked\@gl@checkedmkidx{\gl@xdystring}%

Unsanitize \gl@numberpage, \gl@alphpage, \gl@Alphpage and \gl@romanpage
(thanks to David Carlisle for the suggestion.)

2836 \@for\@gl@tmp:=\gl@protected@pagefmts\do
2837 {%
2838 \edef\@gl@sanitized@tmp{\expandafter\@gobble\string\\expandonce\@gl@tmp}%
2839 \@onelevel@sanitize\@gl@sanitized@tmp
2840 \edef\gl@dostubst{%
2841 \noexpand\DTLsubstituteall\noexpand\gl@xdystring
2842 {\@gl@sanitized@tmp}{\expandonce\@gl@tmp}%
2843 }%
2844 \gl@dostubst
2845 }%

Assign to required control sequence
2846 \let#1=\gl@xdystring
2847 }

```

Catch special characters (argument must be a control sequence):

gl@checkmkidxchars

```

2848 \newcommand{\@gl@checkmkidxchars}[1]{%
2849 \ifgl@xindy
2850 \@gl@escbsdq{#1}%
2851 \else
2852 \def\@gl@checkedmkidx{%
2853 \expandafter\@gl@checkquote#1\@nil""\null
2854 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
2855 \def\@gl@checkedmkidx{%
2856 \expandafter\@gl@checkescquote#1\@nil\""\null
2857 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
2858 \def\@gl@checkedmkidx{%
2859 \expandafter\@gl@checkescactual#1\@nil\?\?\null
2860 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
2861 \def\@gl@checkedmkidx{%
2862 \expandafter\@gl@checkactual#1\@nil??\null
2863 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
2864 \def\@gl@checkedmkidx{%
2865 \expandafter\@gl@checkbar#1\@nil||\null
2866 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
2867 \def\@gl@checkedmkidx{%
2868 \expandafter\@gl@checkescbar#1\@nil\\|\null
2869 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
2870 \def\@gl@checkedmkidx{%
2871 \expandafter\@gl@checklevel#1\@nil!!\null

```

```

2872 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2873 \fi
2874 }

```

Update the control sequence and strip trailing \@nil:

\@gls@updatechecked

```

2875 \def\@gls@updatechecked#1\@nil#2{\def#2{#1}}

```

\@gls@tmpb Define temporary token

```

2876 \newtoks\@gls@tmpb

```

\@gls@checkquote Replace " with "" since " is a makeindex special character.

```

2877 \def\@gls@checkquote#1"#2"#3\null{%
2878 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2879 \toks@={#1}%
2880 \ifx\null#2\null
2881 \ifx\null#3\null
2882 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2883 \def\@gls@checkquote{\relax}%
2884 \else
2885 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2886 \@gls@quotechar\@gls@quotechar\@gls@quotechar\@gls@quotechar}%
2887 \def\@gls@checkquote{\@gls@checkquote#3\null}%
2888 \fi
2889 \else
2890 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2891 \@gls@quotechar\@gls@quotechar}%
2892 \ifx\null#3\null
2893 \def\@gls@checkquote{\@gls@checkquote#2""\null}%
2894 \else
2895 \def\@gls@checkquote{\@gls@checkquote#2"#3\null}%
2896 \fi
2897 \fi
2898 \@gls@checkquote
2899 }

```

\@gls@checkescquote Do the same for \":

```

2900 \def\@gls@checkescquote#1\"#2\"#3\null{%
2901 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2902 \toks@={#1}%
2903 \ifx\null#2\null
2904 \ifx\null#3\null
2905 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2906 \def\@gls@checkescquote{\relax}%
2907 \else
2908 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2909 \@gls@quotechar\string\"@gls@quotechar
2910 \@gls@quotechar\string\"@gls@quotechar}%

```

```

2911 \def\@gls@checkescquote{\@gls@checkescquote#3\null}%
2912 \fi
2913 \else
2914 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2915 \@gls@quotearchar\string"\@gls@quotearchar}%
2916 \ifx\null#3\null
2917 \def\@gls@checkescquote{\@gls@checkescquote#2\""\null}%
2918 \else
2919 \def\@gls@checkescquote{\@gls@checkescquote#2\">#3\null}%
2920 \fi
2921 \fi
2922 \@gls@checkescquote
2923 }

```

`\@gls@checkescactual` Similarly for `\?` (which is replaces `@` as `makeindex`'s special character):

```

2924 \def\@gls@checkescactual#1\?#2\?#3\null{%
2925 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2926 \toks@={#1}%
2927 \ifx\null#2\null
2928 \ifx\null#3\null
2929 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2930 \def\@gls@checkescactual{\relax}%
2931 \else
2932 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2933 \@gls@quotearchar\string"\@gls@actualchar
2934 \@gls@quotearchar\string"\@gls@actualchar}%
2935 \def\@gls@checkescactual{\@gls@checkescactual#3\null}%
2936 \fi
2937 \else
2938 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2939 \@gls@quotearchar\string"\@gls@actualchar}%
2940 \ifx\null#3\null
2941 \def\@gls@checkescactual{\@gls@checkescactual#2\?\?\null}%
2942 \else
2943 \def\@gls@checkescactual{\@gls@checkescactual#2\?#3\null}%
2944 \fi
2945 \fi
2946 \@gls@checkescactual
2947 }

```

`\@gls@checkescbar` Similarly for `\|`:

```

2948 \def\@gls@checkescbar#1\|#2\|#3\null{%
2949 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2950 \toks@={#1}%
2951 \ifx\null#2\null
2952 \ifx\null#3\null
2953 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2954 \def\@gls@checkescbar{\relax}%
2955 \else

```

```

2956 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2957 \@gls@quotearchar\string\"@gls@encapchar
2958 \@gls@quotearchar\string\"@gls@encapchar}%
2959 \def\@@gls@checkescbar{\@gls@checkescbar#3\null}%
2960 \fi
2961 \else
2962 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2963 \@gls@quotearchar\string\"@gls@encapchar}%
2964 \ifx\null#3\null
2965 \def\@@gls@checkescbar{\@gls@checkescbar#2\|\|\null}%
2966 \else
2967 \def\@@gls@checkescbar{\@gls@checkescbar#2\|#3\null}%
2968 \fi
2969 \fi
2970 \@@gls@checkescbar
2971 }

```

\@gls@checkesclevel Similarly for \!:

```

2972 \def\@gls@checkesclevel#1\!#2\!#3\null{%
2973 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2974 \toks@={#1}%
2975 \ifx\null#2\null
2976 \ifx\null#3\null
2977 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2978 \def\@@gls@checkesclevel{\relax}%
2979 \else
2980 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2981 \@gls@quotearchar\string\"@gls@levelchar
2982 \@gls@quotearchar\string\"@gls@levelchar}%
2983 \def\@@gls@checkesclevel{\@gls@checkesclevel#3\null}%
2984 \fi
2985 \else
2986 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2987 \@gls@quotearchar\string\"@gls@levelchar}%
2988 \ifx\null#3\null
2989 \def\@@gls@checkesclevel{\@gls@checkesclevel#2\!\!\null}%
2990 \else
2991 \def\@@gls@checkesclevel{\@gls@checkesclevel#2\!#3\null}%
2992 \fi
2993 \fi
2994 \@@gls@checkesclevel
2995 }

```

\@gls@checkbar and for |:

```

2996 \def\@gls@checkbar#1|#2|#3\null{%
2997 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2998 \toks@={#1}%
2999 \ifx\null#2\null
3000 \ifx\null#3\null

```

```

3001 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
3002 \def\@@gls@checkbar{\relax}%
3003 \else
3004 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3005 \@gls@quotechar\@gls@encapchar\@gls@quotechar\@gls@encapchar}%
3006 \def\@@gls@checkbar{\@gls@checkbar#3\null}%
3007 \fi
3008 \else
3009 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3010 \@gls@quotechar\@gls@encapchar}%
3011 \ifx\null#3\null
3012 \def\@@gls@checkbar{\@gls@checkbar#2||\null}%
3013 \else
3014 \def\@@gls@checkbar{\@gls@checkbar#2|#3\null}%
3015 \fi
3016 \fi
3017 \@@gls@checkbar
3018 }

```

\@gls@checklevel and for !:

```

3019 \def\@gls@checklevel#1!#2!#3\null{%
3020 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
3021 \toks@={#1}%
3022 \ifx\null#2\null
3023 \ifx\null#3\null
3024 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
3025 \def\@@gls@checklevel{\relax}%
3026 \else
3027 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3028 \@gls@quotechar\@gls@levelchar\@gls@quotechar\@gls@levelchar}%
3029 \def\@@gls@checklevel{\@gls@checklevel#3\null}%
3030 \fi
3031 \else
3032 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3033 \@gls@quotechar\@gls@levelchar}%
3034 \ifx\null#3\null
3035 \def\@@gls@checklevel{\@gls@checklevel#2!!\null}%
3036 \else
3037 \def\@@gls@checklevel{\@gls@checklevel#2!#3\null}%
3038 \fi
3039 \fi
3040 \@@gls@checklevel
3041 }

```

\@gls@checkactual and for ?:

```

3042 \def\@gls@checkactual#1?#2?#3\null{%
3043 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
3044 \toks@={#1}%
3045 \ifx\null#2\null

```

```

3046 \ifx\null#3\null
3047 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
3048 \def\@@gls@checkactual{\relax}%
3049 \else
3050 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3051 \@gls@quotechar\@gls@actualchar\@gls@quotechar\@gls@actualchar}%
3052 \def\@@gls@checkactual{\@gls@checkactual#3\null}%
3053 \fi
3054 \else
3055 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3056 \@gls@quotechar\@gls@actualchar}%
3057 \ifx\null#3\null
3058 \def\@@gls@checkactual{\@gls@checkactual#2??\null}%
3059 \else
3060 \def\@@gls@checkactual{\@gls@checkactual#2?#3\null}%
3061 \fi
3062 \fi
3063 \@@gls@checkactual
3064 }

```

\@gls@xdycheckquote As before but for use with xindy

```

3065 \def\@gls@xdycheckquote#1"#2"#3\null{%
3066 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
3067 \toks@={#1}%
3068 \ifx\null#2\null
3069 \ifx\null#3\null
3070 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
3071 \def\@@gls@xdycheckquote{\relax}%
3072 \else
3073 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3074 \string\string\}%
3075 \def\@@gls@xdycheckquote{\@gls@xdycheckquote#3\null}%
3076 \fi
3077 \else
3078 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3079 \string\}%
3080 \ifx\null#3\null
3081 \def\@@gls@xdycheckquote{\@gls@xdycheckquote#2""\null}%
3082 \else
3083 \def\@@gls@xdycheckquote{\@gls@xdycheckquote#2"#3\null}%
3084 \fi
3085 \fi
3086 \@@gls@xdycheckquote
3087 }

```

s@xdycheckbackslash Need to escape all backslashes for xindy. Define command that will define

```

\@gls@xdycheckbackslash
3088 \edef\def\@gls@xdycheckbackslash{%
3089 \noexpand\def\noexpand\@gls@xdycheckbackslash##1\@backslashchar

```

```

3090    ##2\@backslashchar##3\@noexpand\null{%
3091    \noexpand\@gls@tmpb=\noexpand\expandafter
3092    {\noexpand\@gls@checkedmkidx}%
3093    \noexpand\toks@={##1}%
3094    \noexpand\ifx\noexpand\null##2\noexpand\null
3095    \noexpand\ifx\noexpand\null##3\noexpand\null
3096    \noexpand\edef\noexpand\@gls@checkedmkidx{%
3097    \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@}%
3098    \noexpand\def\noexpand\@gls@xdycheckbackslash{\relax}%
3099    \noexpand\else
3100    \noexpand\edef\noexpand\@gls@checkedmkidx{%
3101    \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
3102    \@backslashchar\@backslashchar\@backslashchar\@backslashchar}%
3103    \noexpand\def\noexpand\@gls@xdycheckbackslash{%
3104    \noexpand\@gls@xdycheckbackslash##3\noexpand\null}%
3105    \noexpand\fi
3106    \noexpand\else
3107    \noexpand\edef\noexpand\@gls@checkedmkidx{%
3108    \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
3109    \@backslashchar\@backslashchar}%
3110    \noexpand\ifx\noexpand\null##3\noexpand\null
3111    \noexpand\def\noexpand\@gls@xdycheckbackslash{%
3112    \noexpand\@gls@xdycheckbackslash##2\@backslashchar
3113    \@backslashchar\noexpand\null}%
3114    \noexpand\else
3115    \noexpand\def\noexpand\@gls@xdycheckbackslash{%
3116    \noexpand\@gls@xdycheckbackslash##2\@backslashchar
3117    ##3\noexpand\null}%
3118    \noexpand\fi
3119    \noexpand\fi
3120    \noexpand\@gls@xdycheckbackslash
3121    }%
3122 }

```

Now go ahead and define \@gls@xdycheckbackslash

```

3123 \def@gls@xdycheckbackslash

```

\glsdohypertarget

```

3124 \newlength\gls@tmplen
3125 \newcommand*{\glsdohypertarget}[2]{%
3126   \settoheight{\gls@tmplen}{#2}%
3127   \raisebox{\gls@tmplen}{\hypertarget{#1}{}}#2%
3128 }

```

\glsdohyperlink

```

3129 \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.


```

3130 \ifcsundef{hyperlink}%
3131 {%
3132   \let\@glslink\@secondoftwo
3133 }%
3134 {%
3135   \let\@glslink\glsdohyperlink
3136 }

```

`\@glstarget` If `\hypertarget` is not defined, `\@glstarget` ignores its first argument and just does the second argument, otherwise it is equivalent to `\hypertarget`.

```

3137 \ifcsundef{hypertarget}%
3138 {%
3139   \let\@glstarget\@secondoftwo
3140 }%
3141 {%
3142   \let\@glstarget\glsdohypertarget
3143 }

```

Glossary hyperlinks can be disabled using `\glsdisablehyper` (effect can be localised):

`\glsdisablehyper`

```

3144 \newcommand{\glsdisablehyper}{%
3145   \KV@glslink@hyperfalse
3146   \let\@glslink\@secondoftwo
3147   \let\@glstarget\@secondoftwo
3148 }

```

Glossary hyperlinks can be enabled using `\glsenablehyper` (effect can be localised):

`\glsenablehyper`

```

3149 \newcommand{\glsenablehyper}{%
3150   \KV@glslink@hypertrue
3151   \let\@glslink\glsdohyperlink
3152   \let\@glstarget\glsdohypertarget
3153 }

```

Provide some convenience commands if not already defined:

```

3154 \providecommand{\@firstofthree}[3]{#1}
3155 \providecommand{\@secondofthree}[3]{#2}

```

Syntax:

`\gls[<options>]{<label>}[<insert text>]`

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
3156 \newrobustcmd*{\gls}{\@gls@hyp@opt\@gls}
```

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

```
\@gls
3157 \newcommand*{\@gls}[2] [] {%
3158   \new@ifnextchar[{\@gls@{#1}{#2}}{\@gls@{#1}{#2} []}]%
3159 }
```

`\@gls@` Read in the final optional argument:

```
3160 \def\@gls@#1#2[#3]{%
3161   \glsdoifexists{#2}%
3162   {%
3163     \let\do@gls@link@checkfirsthyper\@gls@link@checkfirsthyper

3164     \let\glsifplural\@secondoftwo
3165     \let\glsupcase\@firstofthree
3166     \let\glscustomtext\@empty
3167     \def\glsinsert{#3}%
```

Determine what the link text should be (this is stored in `\@glo@text`) Note that `\@gls@link` sets `\glstyp`.

```
3168   \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.

```
3169   \@gls@link[#1]{#2}{\@glo@text}%
```

Indicate that this entry has now been used

```
3170   \ifKV@glslink@local
3171     \glslocalunset{#2}%
3172   \else
3173     \glsunset{#2}%
3174   \fi
3175 }%
3176 }
```

`\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`

```
3177 \newrobustcmd*{\Gls}{\@gls@hyp@opt\@Gls}
```

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

```
3178 \newcommand*{\@Gls}[2] [] {%
3179   \new@ifnextchar[{\@Gls@{#1}{#2}}{\@Gls@{#1}{#2} []}%
3180 }
```

`\@Gls@` Read in the final optional argument:

```
3181 \def\@Gls@#1#2[#3]{%
3182   \glsdoifexists{#2}%
3183   {%
3184     \let\do@gls@link@checkfirsthyper\@gls@link@checkfirsthyper

3185     \let\glsifplural\@secondoftwo
3186     \let\glscapscase\@secondofthree
3187     \let\glscustomtext\@empty
3188     \def\glsinsert{#3}%
```

Determine what the link text should be (this is stored in `\@glo@text`) Note that `\@gls@link` sets `\glstype`.

```
3189   \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.

```
3190   \@gls@link[#1]{#2}{\@glo@text}%
```

Indicate that this entry has now been used

```
3191   \ifKV@gls@link@local
3192     \glslocalunset{#2}%
3193   \else
3194     \glsunset{#2}%
3195   \fi
3196 }%
3197 }
```

`\GLS` behaves like `\gls`, but the link text is converted to uppercase:

`\GLS`

```
3198 \newrobustcmd*{\GLS}{\@gls@hyp@opt\@GLS}
```

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

```
3199 \newcommand*{\@GLS}[2] [] {%
3200   \new@ifnextchar[{\@GLS@{#1}{#2}}{\@GLS@{#1}{#2} []}%
3201 }
```

\@GLS@ Read in the final optional argument:

```
3202 \def\@GLS@#1#2[#3] {%
3203   \glsdoifexists{#2}%
3204   {%
3205     \let\do@gl@link@checkfirsthyper\@gl@link@checkfirsthyper
3206     \let\glsifplural\@secondoftwo
3207     \let\glscapscase\@thirdofthree
3208     \let\glscustomtext\@empty
3209     \def\glsinsert{#3}%
3210     \def\@glo@text{\csname gls@\glstype @entryfmt\endcsname}%
3211     Call \@gl@link If footnote package option has been used and the glossary
3212     type is \acronymtype, suppress hyperlink for first use. Likewise if the hyper-
3213     first=false package option is used.
3214     \@gl@link[#1]{#2}{\@glo@text}%
3215     Indicate that this entry has now been used
3216     \ifKV@gl@link@local
3217       \glslocalunset{#2}%
3218     \else
3219       \glsunset{#2}%
3220     \fi
3221   }%
3222 }
```

Determine what the link text should be (this is stored in \@glo@text). Note that \@gl@link sets \glstype.

```
3210 \def\@glo@text{\csname gls@\glstype @entryfmt\endcsname}%
```

Call \@gl@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.

```
3211 \@gl@link[#1]{#2}{\@glo@text}%
```

Indicate that this entry has now been used

```
3212 \ifKV@gl@link@local
3213   \glslocalunset{#2}%
3214 \else
3215   \glsunset{#2}%
3216 \fi
3217 }%
3218 }
```

\glspl behaves in the same way as \gls except it uses the plural form.

\glspl

```
3219 \newrobustcmd*{\glspl}{\@gl@hyp@opt\@glspl}
```

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

```
3220 \newcommand*{\@glspl}[2] [] {%
3221   \new@ifnextchar[{\@glspl@{#1}{#2}}{\@glspl@{#1}{#2} []}%
3222 }
```

\@glspl@ Read in the final optional argument:

```
3223 \def\@glspl@#1#2[#3] {%
3224   \glsdoifexists{#2}%
3225   {%
3226     \let\do@gl@link@checkfirsthyper\@gl@link@checkfirsthyper
```

```

3227 \let\glsifplural\@firstoftwo
3228 \let\glscapscase\@firstofthree
3229 \let\glscustomtext\@empty
3230 \def\glsinsert{#3}%

```

Determine what the link text should be (this is stored in \@glo@text) Note that \@gls@link sets \glstype.

```

3231 \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.

```

3232 \@gls@link[#1]{#2}{\@glo@text}%

```

Indicate that this entry has now been used

```

3233 \ifKV@glslink@local
3234 \glslocalunset{#2}%
3235 \else
3236 \glsunset{#2}%
3237 \fi
3238 }%
3239 }

```

\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

```

3240 \newrobustcmd*{\Glspl}{\@gls@hyp@opt\@Glspl}

```

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

```

3241 \newcommand*{\@Glspl}[2][{}]{%
3242 \new@ifnextchar[\@Glspl@{#1}{#2}]{\@Glspl@{#1}{#2}[]}%
3243 }

```

\@Glspl@ Read in the final optional argument:

```

3244 \def\@Glspl@#1#2[#3]{%
3245 \glsdoifexists{#2}%
3246 {%
3247 \let\do@gls@link@checkfirsthyper\@gls@link@checkfirsthyper

3248 \let\glsifplural\@firstoftwo
3249 \let\glscapscase\@secondofthree
3250 \let\glscustomtext\@empty
3251 \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.

```

3252 \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.

```
3253 \@gls@link[#1]{#2}{\@glo@text}%
```

Indicate that this entry has now been used

```
3254 \ifKV@glslink@local
3255 \glslocalunset{#2}%
3256 \else
3257 \glsunset{#2}%
3258 \fi
3259 }%
3260 }
```

`\GLSp1` behaves like `\glsp1` except that all the link text is converted to uppercase.

`\GLSp1`

```
3261 \newrobustcmd*{\GLSp1}{\@gls@hyp@opt\@GLSp1}
```

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

```
3262 \newcommand*{\@GLSp1}[2][{}]{%
3263 \new@ifnextchar[{\@GLSp1@{#1}{#2}}{\@GLSp1@{#1}{#2}[]}%
3264 }
```

`\@GLSp1` Read in the final optional argument:

```
3265 \def\@GLSp1@#1#2[#3]{%
3266 \glsdoifexists{#2}%
3267 {%
3268 \let\do@gls@link@checkfirsthyper\@gls@link@checkfirsthyper

3269 \let\glsifplural\@firstoftwo
3270 \let\glscapscase\@thirdofthree
3271 \let\glscustomtext\@empty
3272 \def\glsinsert{#3}%
```

Determine what the link text should be (this is stored in `\@glo@text`) Note that `\@gls@link` sets `\glstype`.

```
3273 \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.

```
3274 \@gls@link[#1]{#2}{\@glo@text}%
```

Indicate that this entry has now been used

```
3275 \ifKV@glslink@local
3276 \glslocalunset{#2}%
3277 \else
```

```

3278     \glsunset{#2}%
3279     \fi
3280 }%
3281 }

```

`\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:

```

3282 \newrobustcmd*{\glsdisp}{\@gls@hyp@opt\@glsdisp}

```

Defined the un-starred form.

`\@glsdisp`

```

3283 \newcommand*{\@glsdisp}[3][\%
3284   \glsdoifexists{#2}]{%

3285   \let\do@gls@link@checkfirsthyper\@gls@link@checkfirsthyper

3286   \let\glsifplural\@secondoftwo
3287   \let\glsupcase\@firstofthree
3288   \def\glscustomtext{#3}%
3289   \def\glsinsert{}%

```

Determine what the link text should be (this is stored in `\@glo@text`) Note that `\@gls@link` sets `\glstyp`.

```

3290   \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.

```

3291   \@gls@link[#1]{#2}{\@glo@text}%

```

Indicate that this entry has now been used

```

3292   \ifKV@glslink@local
3293     \glslocalunset{#2}%
3294   \else
3295     \glsunset{#2}%
3296   \fi
3297 }%
3298 }

```

`\@gls@field@link`

```

3299 \newcommand{\@gls@field@link}[3]{%
3300   \glsdoifexists{#2}%
3301   {%
3302     \let\do@gls@link@checkfirsthyper\relax
3303     \@gls@link[#1]{#2}{#3}%
3304   }%
3305 }

```

`\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`

```
3306 \newrobustcmd*{\glstext}{\@gls@hyp@opt\@glstext}
```

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

```
3307 \newcommand*{\@glstext}[2][\%
```

```
3308 \new@ifnextchar[\@glstext@{#1}{#2}]{\@glstext@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3309 \def\@glstext@#1#2[#3]{\%
```

```
3310 \@gls@field@link{#1}{#2}{\glsentrytext{#2}#3}%
```

```
3311 }
```

`\GLStext` behaves like `\glstext` except the text is converted to uppercase.

`\GLStext`

```
3312 \newrobustcmd*{\GLStext}{\@gls@hyp@opt\@GLStext}
```

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

```
3313 \newcommand*{\@GLStext}[2][\%
```

```
3314 \new@ifnextchar[\@GLStext@{#1}{#2}]{\@GLStext@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3315 \def\@GLStext@#1#2[#3]{\%
```

```
3316 \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentrytext{#2}#3}}%
```

```
3317 }
```

`\Glstext` behaves like `\glstext` except that the first letter of the text is converted to uppercase.

`\Glstext`

```
3318 \newrobustcmd*{\Glstext}{\@gls@hyp@opt\@Glstext}
```

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

```
3319 \newcommand*{\@Glstext}[2][\%
```

```
3320 \new@ifnextchar[\@Glstext@{#1}{#2}]{\@Glstext@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3321 \def\@Glstext@#1#2[#3]{\%
```

```
3322 \@gls@field@link{#1}{#2}{\Glsentrytext{#2}#3}%
```

```
3323 }
```

`\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`

```
3324 \newrobustcmd*{\glsfirst}{\@gls@hyp@opt\@glsfirst}
```


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

```
3325 \newcommand*{\@glsfirst}[2] [] {%
3326   \new@ifnextchar[{\@glsfirst@{#1}{#2}}]{\@glsfirst@{#1}{#2} []}}
```

Read in the final optional argument:

```
3327 \def\@glsfirst@#1#2[#3] {%
3328   \@gls@field@link{#1}{#2}{\glsentryfirst{#2}#3}%
3329 }
```

\Glsfirst behaves like \glsfirst except it displays the first letter in up-percase.

\Glsfirst

```
3330 \newrobustcmd*{\Glsfirst}{\@gls@hyp@opt\@Glsfirst}
```

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

```
3331 \newcommand*{\@Glsfirst}[2] [] {%
3332   \new@ifnextchar[{\@Glsfirst@{#1}{#2}}]{\@Glsfirst@{#1}{#2} []}}
```

Read in the final optional argument:

```
3333 \def\@Glsfirst@#1#2[#3] {%
3334   \@gls@field@link{#1}{#2}{\Glsentryfirst{#2}#3}%
3335 }
```

\GLSfirst behaves like \Glsfirst except it displays the text in uppercase.

\GLSfirst

```
3336 \newrobustcmd*{\GLSfirst}{\@gls@hyp@opt\@GLSfirst}
```

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

```
3337 \newcommand*{\@GLSfirst}[2] [] {%
3338   \new@ifnextchar[{\@GLSfirst@{#1}{#2}}]{\@GLSfirst@{#1}{#2} []}}
```

Read in the final optional argument:

```
3339 \def\@GLSfirst@#1#2[#3] {%
3340   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryfirst{#2}#3}}%
3341 }
```

\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

```
3342 \newrobustcmd*{\glsplural}{\@gls@hyp@opt\@glsplural}
```

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

```
3343 \newcommand*{\@glsplural}[2] [] {%
3344   \new@ifnextchar[{\@glsplural@{#1}{#2}}]{\@glsplural@{#1}{#2} []}}
```

Read in the final optional argument:

```
3345 \def\@glsplural@#1#2[#3]{%
3346   \@gls@field@link{#1}{#2}{\glsentryplural{#2}#3}%
3347 }
```

\Glsplural behaves like \glsplural except that the first letter is converted to uppercase.

\Glsplural

```
3348 \newrobustcmd*{\Glsplural}{\@gls@hyp@opt\@Glsplural}
```

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

```
3349 \newcommand*{\@Glsplural}[2][\%
3350   \new@ifnextchar[{\@Glsplural@{#1}{#2}}{\@Glsplural@{#1}{#2}}{ }]
```

Read in the final optional argument:

```
3351 \def\@Glsplural@#1#2[#3]{%
3352   \@gls@field@link{#1}{#2}{\glsentryplural{#2}#3}%
3353 }
```

\Glsplural behaves like \glsplural except that the text is converted to uppercase.

\GLSplural

```
3354 \newrobustcmd*{\GLSplural}{\@gls@hyp@opt\@GLSplural}
```

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

```
3355 \newcommand*{\@GLSplural}[2][\%
3356   \new@ifnextchar[{\@GLSplural@{#1}{#2}}{\@GLSplural@{#1}{#2}}{ }]
```

Read in the final optional argument:

```
3357 \def\@GLSplural@#1#2[#3]{%
3358   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryplural{#2}#3}}%
3359 }
```

\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

```
3360 \newrobustcmd*{\glsfirstplural}{\@gls@hyp@opt\@glsfirstplural}
```

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

```
3361 \newcommand*{\@glsfirstplural}[2][\%
3362   \new@ifnextchar[{\@glsfirstplural@{#1}{#2}}{\@glsfirstplural@{#1}{#2}}{ }]
```

Read in the final optional argument:

```
3363 \def\@glsfirstplural@#1#2[#3]{%
3364   \@gls@field@link{#1}{#2}{\glsentryfirstplural{#2}#3}%
3365 }
```

`\Glsfirstplural` behaves like `\glsfirstplural` except that the first letter is converted to uppercase.

`\Glsfirstplural`

```
3366 \newrobustcmd*{\Glsfirstplural}{\@gls@hyp@opt\@Glsfirstplural}
```

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

```
3367 \newcommand*{\@Glsfirstplural}[2][\@Glsfirstplural@{#1}{#2}]{\@Glsfirstplural@{#1}{#2}[]}}
```

```
3368 \new@ifnextchar[\@Glsfirstplural@{#1}{#2}]{\@Glsfirstplural@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3369 \def\@Glsfirstplural@#1#2[#3]{\@Glsfirstplural@{#1}{#2}[\@Glsfirstplural@{#1}{#2}[]]}
```

```
3370 \@Glsfirstplural@{#1}{#2}[\@Glsfirstplural@{#1}{#2}[]]}
```

```
3371 }
```

`\Glsfirstplural` behaves like `\glsfirstplural` except that the link text is converted to uppercase.

`\Glsfirstplural`

```
3372 \newrobustcmd*{\Glsfirstplural}{\@gls@hyp@opt\@Glsfirstplural}
```

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

```
3373 \newcommand*{\@Glsfirstplural}[2][\@Glsfirstplural@{#1}{#2}]{\@Glsfirstplural@{#1}{#2}[]}}
```

```
3374 \new@ifnextchar[\@Glsfirstplural@{#1}{#2}]{\@Glsfirstplural@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3375 \def\@Glsfirstplural@#1#2[#3]{\@Glsfirstplural@{#1}{#2}[\@Glsfirstplural@{#1}{#2}[]]}
```

```
3376 \@Glsfirstplural@{#1}{#2}[\@Glsfirstplural@{#1}{#2}[]]}
```

```
3377 }
```

`\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`

```
3378 \newrobustcmd*{\glsname}{\@gls@hyp@opt\@glsname}
```

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

```
3379 \newcommand*{\@glsname}[2][\@glsname@{#1}{#2}]{\@glsname@{#1}{#2}[]}}
```

```
3380 \new@ifnextchar[\@glsname@{#1}{#2}]{\@glsname@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3381 \def\@glsname@#1#2[#3]{\@glsname@{#1}{#2}[\@glsname@{#1}{#2}[]]}
```

```
3382 \@glsname@{#1}{#2}[\@glsname@{#1}{#2}[]]}
```

```
3383 }
```

`\Glsname` behaves like `\glsname` except that the first letter is converted to uppercase.

`\Glsname`

```
3384 \newrobustcmd*{\Glsname}{\@gls@hyp@opt\@Glsname}
```

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

```
3385 \newcommand*{\@Glsname}[2] [] {%
3386   \new@ifnextchar[{\@Glsname@{#1}{#2}}{\@Glsname@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3387 \def\@Glsname@#1#2[#3] {%
3388   \@gls@field@link{#1}{#2}{\Glsentryname{#2}#3}%
3389 }
```

\GLSname behaves like \glsname except that the link text is converted to uppercase.

\GLSname

```
3390 \newrobustcmd*{\GLSname}{\@gls@hyp@opt\@GLSname}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3391 \newcommand*{\@GLSname}[2] [] {%
3392   \new@ifnextchar[{\@GLSname@{#1}{#2}}{\@GLSname@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3393 \def\@GLSname@#1#2[#3] {%
3394   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\Glsentryname{#2}#3}}%
3395 }
```

\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

```
3396 \newrobustcmd*{\glsdesc}{\@gls@hyp@opt\@glsdesc}
```

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

```
3397 \newcommand*{\@glsdesc}[2] [] {%
3398   \new@ifnextchar[{\@glsdesc@{#1}{#2}}{\@glsdesc@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3399 \def\@glsdesc@#1#2[#3] {%
3400   \@gls@field@link{#1}{#2}{\Glsentrydesc{#2}#3}%
3401 }
```

\Glsdesc behaves like \glsdesc except that the first letter is converted to uppercase.

\Glsdesc

```
3402 \newrobustcmd*{\Glsdesc}{\@gls@hyp@opt\@Glsdesc}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3403 \newcommand*{\@Glsdesc}[2] [] {%
3404   \new@ifnextchar[{\@Glsdesc@{#1}{#2}}{\@Glsdesc@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3405 \def\@Glsdesc@#1#2[#3]{%  
3406   \@gls@field@link{#1}{#2}{\Glsentrydesc{#2}#3}%  
3407 }
```

\Glsdesc behaves like \glsdesc except that the link text is converted to uppercase.

\Glsdesc

```
3408 \newrobustcmd*{\Glsdesc}{\@gls@hyp@opt\@Glsdesc}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3409 \newcommand*{\@Glsdesc}[2][ ]{%  
3410   \new@ifnextchar[{\@Glsdesc@{#1}{#2}}{\@Glsdesc@{#1}{#2}[ ]}]
```

Read in the final optional argument:

```
3411 \def\@Glsdesc@#1#2[#3]{%  
3412   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\Glsentrydesc{#2}#3}}%  
3413 }
```

\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

```
3414 \newrobustcmd*{\glsdescplural}{\@gls@hyp@opt\@glsdescplural}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3415 \newcommand*{\@glsdescplural}[2][ ]{%  
3416   \new@ifnextchar[{\@glsdescplural@{#1}{#2}}{\@glsdescplural@{#1}{#2}[ ]}]
```

Read in the final optional argument:

```
3417 \def\@glsdescplural@#1#2[#3]{%  
3418   \@gls@field@link{#1}{#2}{\Glsentrydescplural{#2}#3}%  
3419 }
```

\Glsdescplural behaves like \glsdescplural except that the first letter is converted to uppercase.

\Glsdescplural

```
3420 \newrobustcmd*{\Glsdescplural}{\@gls@hyp@opt\@Glsdescplural}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3421 \newcommand*{\@Glsdescplural}[2][ ]{%  
3422   \new@ifnextchar[{\@Glsdescplural@{#1}{#2}}{\@Glsdescplural@{#1}{#2}[ ]}]
```

Read in the final optional argument:

```
3423 \def\@Glsdescplural@#1#2[#3]{%  
3424   \@gls@field@link{#1}{#2}{\Glsentrydescplural{#2}#3}%  
3425 }
```

`\GLSdescplural` behaves like `\glsdescplural` except that the link text is converted to uppercase.

`\GLSdescplural`

```
3426 \newrobustcmd*{\GLSdescplural}{\@gls@hyp@opt\@GLSdescplural}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3427 \newcommand*{\@GLSdescplural}[2] [] {%
```

```
3428   \new@ifnextchar[{\@GLSdescplural@{#1}{#2}}{\@GLSdescplural@{#1}{#2} []}]
```

Read in the final optional argument:

```
3429 \def\@GLSdescplural@#1#2[#3] {%
```

```
3430   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentrydescplural{#2}{#3}}}%
```

```
3431 }
```

`\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`

```
3432 \newrobustcmd*{\glssymbol}{\@gls@hyp@opt\@glssymbol}
```

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

```
3433 \newcommand*{\@glssymbol}[2] [] {%
```

```
3434   \new@ifnextchar[{\@glssymbol@{#1}{#2}}{\@glssymbol@{#1}{#2} []}]
```

Read in the final optional argument:

```
3435 \def\@glssymbol@#1#2[#3] {%
```

```
3436   \@gls@field@link{#1}{#2}{\glsentrysymbol{#2}{#3}}%
```

```
3437 }
```

`\Glsymbol` behaves like `\glssymbol` except that the first letter is converted to uppercase.

`\Glsymbol`

```
3438 \newrobustcmd*{\Glsymbol}{\@gls@hyp@opt\@Glsymbol}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3439 \newcommand*{\@Glsymbol}[2] [] {%
```

```
3440   \new@ifnextchar[{\@Glsymbol@{#1}{#2}}{\@Glsymbol@{#1}{#2} []}]
```

Read in the final optional argument:

```
3441 \def\@Glsymbol@#1#2[#3] {%
```

```
3442   \@gls@field@link{#1}{#2}{\Glsentrysymbol{#2}{#3}}%
```

```
3443 }
```

`\GLSsymbol` behaves like `\glssymbol` except that the link text is converted to uppercase.

`\GLSsymbol`

```
3444 \newrobustcmd*{\GLSsymbol}{\@gls@hyp@opt\@GLSsymbol}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3445 \newcommand*{\@GLSsymbol}[2] [] {%
3446   \new@ifnextchar[{\@GLSsymbol@{#1}{#2}}{\@GLSsymbol@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3447 \def\@GLSsymbol@#1#2[#3] {%
3448   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentrysymbol{#2}#3}}%
3449 }
```

`\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`

```
3450 \newrobustcmd*{\glsymbolplural}{\@gls@hyp@opt\@glsymbolplural}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3451 \newcommand*{\@glsymbolplural}[2] [] {%
3452   \new@ifnextchar[{\@glsymbolplural@{#1}{#2}}{\@glsymbolplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3453 \def\@glsymbolplural@#1#2[#3] {%
3454   \@gls@field@link{#1}{#2}{\glsentrysymbolplural{#2}#3}%
3455 }
```

`\Glsymbolplural` behaves like `\glsymbolplural` except that the first letter is converted to uppercase.

`\Glsymbolplural`

```
3456 \newrobustcmd*{\Glsymbolplural}{\@gls@hyp@opt\@Glsymbolplural}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3457 \newcommand*{\@Glsymbolplural}[2] [] {%
3458   \new@ifnextchar[{\@Glsymbolplural@{#1}{#2}}{\@Glsymbolplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3459 \def\@Glsymbolplural@#1#2[#3] {%
3460   \@gls@field@link{#1}{#2}{\Glsentrysymbolplural{#2}#3}%
3461 }
```

`\GLSsymbolplural` behaves like `\glsymbolplural` except that the link text is converted to uppercase.

`\GLSsymbolplural`

```
3462 \newrobustcmd*{\GLSsymbolplural}{\@gls@hyp@opt\@GLSsymbolplural}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3463 \newcommand*{\@GLSsymbolplural}[2] [] {%
3464   \new@ifnextchar[{\@GLSsymbolplural@{#1}{#2}}{\@GLSsymbolplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3465 \def\@GLSsymbolplural@#1#2[#3]{%
3466   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentrysymbolplural{#2}#3}}%
3467 }
```

`\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`

```
3468 \newrobustcmd*{\glsuseri}{\@gls@hyp@opt\@glsuseri}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3469 \newcommand*{\@glsuseri}[2] [] {%
3470   \new@ifnextchar[{\@glsuseri@{#1}{#2}}{\@glsuseri@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3471 \def\@glsuseri@#1#2[#3]{%
3472   \@gls@field@link{#1}{#2}{\glsentryuseri{#2}#3}%
3473 }
```

`\Glsuseri` behaves like `\glsuseri` except that the first letter is converted to uppercase.

`\Glsuseri`

```
3474 \newrobustcmd*{\Glsuseri}{\@gls@hyp@opt\@Glsuseri}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3475 \newcommand*{\@Glsuseri}[2] [] {%
3476   \new@ifnextchar[{\@Glsuseri@{#1}{#2}}{\@Glsuseri@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3477 \def\@Glsuseri@#1#2[#3]{%
3478   \@gls@field@link{#1}{#2}{\Glsentryuseri{#2}#3}%
3479 }
```

`\GLSuseri` behaves like `\glsuseri` except that the link text is converted to uppercase.

`\GLSuseri`

```
3480 \newrobustcmd*{\GLSuseri}{\@gls@hyp@opt\@GLSuseri}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3481 \newcommand*{\@GLSuseri}[2] [] {%
3482   \new@ifnextchar[{\@GLSuseri@{#1}{#2}}{\@GLSuseri@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3483 \def\@GLSuseri@#1#2[#3]{%
3484   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryuseri{#2}#3}}%
3485 }
```


`\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`

```
3486 \newrobustcmd*{\glsuserii}{\@gls@hyp@opt\@glsuserii}
```

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

```
3487 \newcommand*{\@glsuserii}[2] [] {%
```

```
3488   \new@ifnextchar[{\@glsuserii@{#1}{#2}}{\@glsuserii@{#1}{#2} []}]
```

Read in the final optional argument:

```
3489 \def\@glsuserii@#1#2[#3] {%
```

```
3490   \@gls@field@link{#1}{#2}{\glsentryuserii{#2}#3}%
```

```
3491 }
```

`\Glsuserii` behaves like `\glsuserii` except that the first letter is converted to uppercase.

`\Glsuserii`

```
3492 \newrobustcmd*{\Glsuserii}{\@gls@hyp@opt\@Glsuserii}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3493 \newcommand*{\@Glsuserii}[2] [] {%
```

```
3494   \new@ifnextchar[{\@Glsuserii@{#1}{#2}}{\@Glsuserii@{#1}{#2} []}]
```

Read in the final optional argument:

```
3495 \def\@Glsuserii@#1#2[#3] {%
```

```
3496   \@gls@field@link{#1}{#2}{\Glsentryuserii{#2}#3}%
```

```
3497 }
```

`\GLSuserii` behaves like `\glsuserii` except that the link text is converted to uppercase.

`\GLSuserii`

```
3498 \newrobustcmd*{\GLSuserii}{\@gls@hyp@opt\@GLSuserii}
```

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

```
3499 \newcommand*{\@GLSuserii}[2] [] {%
```

```
3500   \new@ifnextchar[{\@GLSuserii@{#1}{#2}}{\@GLSuserii@{#1}{#2} []}]
```

Read in the final optional argument:

```
3501 \def\@GLSuserii@#1#2[#3] {%
```

```
3502   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryuserii{#2}#3}}%
```

```
3503 }
```

`\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`

```
3504 \newrobustcmd*{\glsuseriii}{\@gls@hyp@opt\@glsuseriii}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3505 \newcommand*{\@glsuseriii}[2] [] {%
3506   \new@ifnextchar[{\@glsuseriii@{#1}{#2}}{\@glsuseriii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3507 \def\@glsuseriii@#1#2[#3]{%
3508   \@gls@field@link{#1}{#2}{\glstentryuseriii{#2}#3}%
3509 }
```

\Glsuseriii behaves like \glsuseriii except that the first letter is converted to uppercase.

\Glsuseriii

```
3510 \newrobustcmd*{\Glsuseriii}{\@gls@hyp@opt\@Glsuseriii}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3511 \newcommand*{\@Glsuseriii}[2] [] {%
3512   \new@ifnextchar[{\@Glsuseriii@{#1}{#2}}{\@Glsuseriii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3513 \def\@Glsuseriii@#1#2[#3]{%
3514   \@gls@field@link{#1}{#2}{\Glstentryuseriii{#2}#3}%
3515 }
```

\GLSuseriii behaves like \glsuseriii except that the link text is converted to uppercase.

\GLSuseriii

```
3516 \newrobustcmd*{\GLSuseriii}{\@gls@hyp@opt\@GLSuseriii}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3517 \newcommand*{\@GLSuseriii}[2] [] {%
3518   \new@ifnextchar[{\@GLSuseriii@{#1}{#2}}{\@GLSuseriii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3519 \def\@GLSuseriii@#1#2[#3]{%
3520   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glstentryuseriii{#2}#3}}%
3521 }
```

\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

```
3522 \newrobustcmd*{\glsuseriv}{\@gls@hyp@opt\@glsuseriv}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3523 \newcommand*{\@glsuseriv}[2] [] {%
3524   \new@ifnextchar[{\@glsuseriv@{#1}{#2}}{\@glsuseriv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3525 \def\@glsuseriv@#1#2[#3]{%
3526   \@gls@field@link{#1}{#2}{\glstentryuseriv{#2}#3}%
3527 }
```

`\Glsuseriv` behaves like `\glsuseriv` except that the first letter is converted to uppercase.

`\Glsuseriv`

```
3528 \newrobustcmd*{\Glsuseriv}{\@gls@hyp@opt\@Glsuseriv}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3529 \newcommand*{\@Glsuseriv}[2][]{%
3530   \new@ifnextchar[{\@Glsuseriv@{#1}{#2}}{\@Glsuseriv@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3531 \def\@Glsuseriv@#1#2[#3]{%
3532   \@gls@field@link{#1}{#2}{\glstentryuseriv{#2}#3}%
3533 }
```

`\GLSuseriv` behaves like `\glsuseriv` except that the link text is converted to uppercase.

`\GLSuseriv`

```
3534 \newrobustcmd*{\GLSuseriv}{\@gls@hyp@opt\@GLSuseriv}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3535 \newcommand*{\@GLSuseriv}[2][]{%
3536   \new@ifnextchar[{\@GLSuseriv@{#1}{#2}}{\@GLSuseriv@{#1}{#2}[]}]
```

Read in the final optional argument:

```
3537 \def\@GLSuseriv@#1#2[#3]{%
3538   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glstentryuseriv{#2}#3}}%
3539 }
```

`\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`

```
3540 \newrobustcmd*{\glsuserv}{\@gls@hyp@opt\@glsuserv}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3541 \newcommand*{\@glsuserv}[2][]{%
3542   \new@ifnextchar[{\@glsuserv@{#1}{#2}}{\@glsuserv@{#1}{#2}[]}]
```

Read in the final optional argument:

```
3543 \def\@glsuserv@#1#2[#3]{%
3544   \@gls@field@link{#1}{#2}{\glstentryuserv{#2}#3}%
3545 }
```

`\Glsuserv` behaves like `\glsuserv` except that the first letter is converted to uppercase.

`\Glsuserv`

```
3546 \newrobustcmd*{\Glsuserv}{\@gls@hyp@opt\@Glsuserv}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3547 \newcommand*{\@Glsuserv}[2][\@Glsuserv]{\@Glsuserv@{#1}{#2}}{\@Glsuserv@{#1}{#2}}}
```

```
3548 \new@ifnextchar[\@Glsuserv@{#1}{#2}}{\@Glsuserv@{#1}{#2}}}
```

Read in the final optional argument:

```
3549 \def\@Glsuserv@#1#2[#3]{\@Glsuserv@{#1}{#2}{\@Glsuserv@{#1}{#2}}}
```

```
3550 \@Glsuserv@{#1}{#2}{\@Glsuserv@{#1}{#2}}}
```

```
3551 }
```

`\GLSuserv` behaves like `\glsuserv` except that the link text is converted to uppercase.

`\GLSuserv`

```
3552 \newrobustcmd*{\GLSuserv}{\@gls@hyp@opt\@GLSuserv}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3553 \newcommand*{\@GLSuserv}[2][\@GLSuserv]{\@GLSuserv@{#1}{#2}}{\@GLSuserv@{#1}{#2}}}
```

```
3554 \new@ifnextchar[\@GLSuserv@{#1}{#2}}{\@GLSuserv@{#1}{#2}}}
```

Read in the final optional argument:

```
3555 \def\@GLSuserv@#1#2[#3]{\@GLSuserv@{#1}{#2}{\@GLSuserv@{#1}{#2}}}
```

```
3556 \@GLSuserv@{#1}{#2}{\@GLSuserv@{#1}{#2}}}
```

```
3557 }
```

`\glsuservi` behaves like `\gls` except it always uses the value given by the `user6` key and it doesn't mark the entry as used.

`\glsuservi`

```
3558 \newrobustcmd*{\glsuservi}{\@gls@hyp@opt\@glsuservi}
```

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

```
3559 \newcommand*{\@glsuservi}[2][\@glsuservi]{\@glsuservi@{#1}{#2}}{\@glsuservi@{#1}{#2}}}
```

```
3560 \new@ifnextchar[\@glsuservi@{#1}{#2}}{\@glsuservi@{#1}{#2}}}
```

Read in the final optional argument:

```
3561 \def\@glsuservi@#1#2[#3]{\@glsuservi@{#1}{#2}{\@glsuservi@{#1}{#2}}}
```

```
3562 \@glsuservi@{#1}{#2}{\@glsuservi@{#1}{#2}}}
```

```
3563 }
```

`\Glsuservi` behaves like `\glsuservi` except that the first letter is converted to uppercase.

`\Glsuservi`

```
3564 \newrobustcmd*{\Glsuservi}{\@gls@hyp@opt\@Glsuservi}
```

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

```
3565 \newcommand*{\@Glsuservi}[2] [] {%
3566   \new@ifnextchar[{\@Glsuservi@{#1}{#2}}{\@Glsuservi@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3567 \def\@Glsuservi@#1#2[#3] {%
3568   \@gls@field@link{#1}{#2}{\glentryuservi{#2}#3}%
3569 }
```

\GLSuservi behaves like \glsuservi except that the link text is converted to uppercase.

\GLSuservi

```
3570 \newrobustcmd*{\GLSuservi}{\@gls@hyp@opt\@GLSuservi}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3571 \newcommand*{\@GLSuservi}[2] [] {%
3572   \new@ifnextchar[{\@GLSuservi@{#1}{#2}}{\@GLSuservi@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3573 \def\@GLSuservi@#1#2[#3] {%
3574   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glentryuservi{#2}#3}}%
3575 }
```

Now deal with acronym related keys. First the short form:

\acrshort

```
3576 \newrobustcmd*{\acrshort}{\@gls@hyp@opt\@ns@acrshort}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3577 \newcommand*{\@ns@acrshort}[2] [] {%
3578   \new@ifnextchar[{\@acrshort{#1}{#2}}{\@acrshort{#1}{#2} []}]%
3579 }
```

Read in the final optional argument:

```
3580 \def\@acrshort#1#2[#3] {%
3581   \glsdoifexists{#2}%
3582   {%
3583     \let\do@gls@link@checkfirsthyper\relax
3584     \let\glsifplural\@secondoftwo
3585     \let\glscapscase\@firstofthree
3586     \let\glsinsert\@empty
3587     \def\glscustomtext{%
3588       \acronymfont{\glentryshort{#2}}#3%
3589     }%
```

Call \@gls@link Note that \@gls@link sets \glstype.

```
3590 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3591 }%
3592 }
```

\Acrshort

```
3593 \newrobustcmd*{\Acrshort}{\@gls@hyp@opt\@ns@Acrshort}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3594 \newcommand*{\ns@Acrshort}[2] [] {%
3595 \new@ifnextchar[{\@Acrshort{#1}{#2}}{\@Acrshort{#1}{#2} []}%
3596 }
```

Read in the final optional argument:

```
3597 \def\@Acrshort#1#2[#3] {%
3598 \glsdoifexists{#2}%
3599 {%
3600 \let\do@gls@link@checkfirsthyper\relax

3601 \def\glslabel{#2}%
3602 \let\glsifplural\@secondoftwo
3603 \let\glsapscase\@secondofthree
3604 \let\glsinsert\@empty
3605 \def\glscustomtext{%
3606 \acronymfont{\Glsentryshort{#2}}#3%
3607 }%
```

Call \@gls@link Note that \@gls@link sets \glstype.

```
3608 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3609 }%
3610 }
```

\ACRshort

```
3611 \newrobustcmd*{\ACRshort}{\@gls@hyp@opt\@ns@ACRshort}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3612 \newcommand*{\ns@ACRshort}[2] [] {%
3613 \new@ifnextchar[{\@ACRshort{#1}{#2}}{\@ACRshort{#1}{#2} []}%
3614 }
```

Read in the final optional argument:

```
3615 \def\@ACRshort#1#2[#3] {%
3616 \glsdoifexists{#2}%
3617 {%
3618 \let\do@gls@link@checkfirsthyper\relax
```

```

3619 \def\glslabel{#2}%
3620 \let\glsifplural\@secondoftwo
3621 \let\glscapscase\@thirdofthree
3622 \let\glsinsert\@empty
3623 \def\glscustomtext{%
3624     \mfirstucMakeUppercase{\acronymfont{\glsentryshort{#2}}#3}%
3625 }%

```

Call \@gls@link Note that \@gls@link sets \glstype.

```

3626 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3627 }%
3628 }

```

Short plural:

\acrshortpl

```

3629 \newrobustcmd*{\acrshortpl}{\@gls@hyp@opt\@ns@acrshortpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

3630 \newcommand*{\ns@acrshortpl}[2] [] {%
3631     \new@ifnextchar[{\acrshortpl{#1}{#2}}{\acrshortpl{#1}{#2} []}%
3632 }

```

Read in the final optional argument:

```

3633 \def\@acrshortpl#1#2[#3] {%
3634     \glsdoifexists{#2}%
3635     {%
3636         \let\do@gls@link@checkfirsthyper\relax
3637
3638         \def\glslabel{#2}%
3639         \let\glsifplural\@firstoftwo
3640         \let\glscapscase\@firstofthree
3641         \let\glsinsert\@empty
3642         \def\glscustomtext{%
3643             \acronymfont{\glsentryshortpl{#2}}#3%
3644         }%
3645     }%
3646 }

```

Call \@gls@link Note that \@gls@link sets \glstype.

```

3644 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3645 }%
3646 }

```

\Acrshortpl

```

3647 \newrobustcmd*{\Acrshortpl}{\@gls@hyp@opt\@ns@Acrshortpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

3648 \newcommand*{\ns@Acrshortpl}[2] [] {%
3649     \new@ifnextchar[{\Acrshortpl{#1}{#2}}{\Acrshortpl{#1}{#2} []}%
3650 }

```

Read in the final optional argument:

```
3651 \def\@Acrshortpl#1#2[#3]{%
3652   \glsdoifexists{#2}%
3653   {%
3654     \let\do@gl@link@checkfirsthyper\relax

3655     \def\glslabel{#2}%
3656     \let\glsifplural\@firstoftwo
3657     \let\glscapscase\@secondofthree
3658     \let\glsinsert\@empty
3659     \def\glscustomtext{%
3660       \acronymfont{\glentryshortpl{#2}}#3%
3661     }%
```

Call \@gl@link Note that \@gl@link sets \glstype.

```
3662   \@gl@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
3663   }%
3664 }
```

\ACRshortpl

```
3665 \newrobustcmd*{\ACRshortpl}{\@gl@hyp@opt\ns@ACRshortpl}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3666 \newcommand*{\ns@ACRshortpl}[2][ ]{%
3667   \new@ifnextchar[{\@ACRshortpl{#1}{#2}}{\@ACRshortpl{#1}{#2}[]}%
3668 }
```

Read in the final optional argument:

```
3669 \def\@ACRshortpl#1#2[#3]{%
3670   \glsdoifexists{#2}%
3671   {%
3672     \let\do@gl@link@checkfirsthyper\relax

3673     \def\glslabel{#2}%
3674     \let\glsifplural\@firstoftwo
3675     \let\glscapscase\@thirdofthree
3676     \let\glsinsert\@empty
3677     \def\glscustomtext{%
3678       \mfirstucMakeUppercase{\acronymfont{\glentryshortpl{#2}}#3}%
3679     }%
```

Call \@gl@link Note that \@gl@link sets \glstype.

```
3680   \@gl@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
3681   }%
3682 }
```

\acrlong

```
3683 \newrobustcmd*{\acrlong}{\@gl@hyp@opt\ns@acrlong}
```


Define the un-starred form. Need to determine if there is a final optional argument

```
3684 \newcommand*{\ns@acrlong}[2] [] {%
3685   \new@ifnextchar[{\@acrlong{#1}{#2}}{\@acrlong{#1}{#2} []}%
3686 }
```

Read in the final optional argument:

```
3687 \def\@acrlong#1#2[#3] {%
3688   \glstoifexists{#2}%
3689   {%
3690     \let\do@gl@link@checkfirsthyper\relax

3691     \def\glslabel{#2}%
3692     \let\gl@ifplural\@secondoftwo
3693     \let\glscapscase\@firstofthree
3694     \let\glinsert\@empty
```

Bug fix v4.02 removed \acronymfont from \glscustomtext (\acronymfont only designed for short form).

```
3695   \def\glscustomtext{%
3696     \glstrylong{#2}#3%
3697   }%
```

Call \@gl@link Note that \@gl@link sets \glstype.

```
3698   \@gl@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
3699   }%
3700 }
```

\Acrlong

```
3701 \newrobustcmd*{\Acrlong}{\@gl@hyp@opt\ns@Acrlong}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3702 \newcommand*{\ns@Acrlong}[2] [] {%
3703   \new@ifnextchar[{\@Acrlong{#1}{#2}}{\@Acrlong{#1}{#2} []}%
3704 }
```

Read in the final optional argument:

```
3705 \def\@Acrlong#1#2[#3] {%
3706   \glstoifexists{#2}%
3707   {%
3708     \let\do@gl@link@checkfirsthyper\relax

3709     \def\glslabel{#2}%
3710     \let\gl@ifplural\@secondoftwo
3711     \let\glscapscase\@secondofthree
3712     \let\glinsert\@empty
```

Bug fix v4.02 removed \acronymfont from \glscustomtext (\acronymfont only designed for short form).

```

3713 \def\glscustomtext{%
3714 \Glsentrylong{#2}#3%
3715 }%

```

Call \@gls@link. Note that \@gls@link sets \glstype.

```

3716 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3717 }%
3718 }

```

\ACRlong

```

3719 \newrobustcmd*{\ACRlong}{\@gls@hyp@opt\@ns@ACRlong}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

3720 \newcommand*{\ns@ACRlong}[2][{}]{%
3721 \new@ifnextchar[{\@ACRlong{#1}{#2}}{\@ACRlong{#1}{#2}[]}%
3722 }

```

Read in the final optional argument:

```

3723 \def\@ACRlong#1#2[#3]{%
3724 \glsdoifexists{#2}%
3725 {%
3726 \let\do@gls@link@checkfirsthyper\relax
3727 \def\glslabel{#2}%
3728 \let\glsifplural\@secondoftwo
3729 \let\glscapscase\@thirdofthree
3730 \let\glsinsert\@empty

```

Bug fix v4.02 removed \acronymfont from \glscustomtext (\acronymfont only designed for short form).

```

3731 \def\glscustomtext{%
3732 \mfirstucMakeUppercase{\glsentrylong{#2}#3}%
3733 }%

```

Call \@gls@link. Note that \@gls@link sets \glstype.

```

3734 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3735 }%
3736 }

```

Short plural:

\acrlongpl

```

3737 \newrobustcmd*{\acrlongpl}{\@gls@hyp@opt\@ns@acrlongpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

3738 \newcommand*{\ns@acrlongpl}[2][{}]{%
3739 \new@ifnextchar[{\@acrlongpl{#1}{#2}}{\@acrlongpl{#1}{#2}[]}%
3740 }

```

Read in the final optional argument:

```

3741 \def\@acrlongpl#1#2[#3]{%
3742   \glsdoifexists{#2}%
3743   {%
3744     \let\do@gl@link@checkfirsthyper\relax

3745   \def\glslabel{#2}%
3746   \let\glsifplural\@firstoftwo
3747   \let\glscapscase\@firstofthree
3748   \let\glsinsert\@empty

```

Bug fix v4.02 removed \acronymfont from \glscustomtext (\acronymfont only designed for short form).

```

3749   \def\glscustomtext{%
3750     \glentrylongpl{#2}#3%
3751   }%

```

Call \@gl@link. Note that \@gl@link sets \glstype.

```

3752   \@gl@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
3753   }%
3754 }

```

\Acrlongpl

```

3755 \newrobustcmd*{\Acrlongpl}{\@gl@hyp@opt\@ns@Acrlongpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

3756 \newcommand*{\@ns@Acrlongpl}[2][]{%
3757   \new@ifnextchar[{\@Acrlongpl{#1}{#2}}{\@Acrlongpl{#1}{#2}[]}%
3758 }

```

Read in the final optional argument:

```

3759 \def\@Acrlongpl#1#2[#3]{%
3760   \glsdoifexists{#2}%
3761   {%
3762     \let\do@gl@link@checkfirsthyper\relax

3763   \def\glslabel{#2}%
3764   \let\glsifplural\@firstoftwo
3765   \let\glscapscase\@secondofthree
3766   \let\glsinsert\@empty

```

Bug fix v4.02 removed \acronymfont from \glscustomtext (\acronymfont only designed for short form).

```

3767   \def\glscustomtext{%
3768     \Glsentrylongpl{#2}#3%
3769   }%

```

Call \@gl@link. Note that \@gl@link sets \glstype.

```

3770   \@gl@link[#1]{#2}{\csname gls@gl@type @entryfmt\endcsname}%
3771   }%
3772 }

```

\ACRlongpl

```
3773 \newrobustcmd*{\ACRlongpl}{\@gls@hyp@opt\ns@ACRlongpl}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3774 \newcommand*{\ns@ACRlongpl}[2] [] {%
3775   \new@ifnextchar[{\@ACRlongpl{#1}{#2}}{\@ACRlongpl{#1}{#2} []}%
3776 }
```

Read in the final optional argument:

```
3777 \def\@ACRlongpl#1#2[#3] {%
3778   \glsdoifexists{#2}%
3779   {%
3780     \let\do@gls@link@checkfirsthyper\relax
3781     \def\glslabel{#2}%
3782     \let\glsifplural\@firstoftwo
3783     \let\glscapscase\@thirdofthree
3784     \let\glsinsert\@empty
```

Bug fix v4.02 removed \acronymfont from \glscustomtext (\acronymfont only designed for short form).

```
3785   \def\glscustomtext{%
3786     \mfirstucMakeUppercase{\glsentrylongpl{#2}{#3}%
3787   }%
```

Call \@gls@link. Note that \@gls@link sets \glstype.

```
3788   \@gls@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
3789   }%
3790 }
```

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{\langle label \rangle}{\langle field \rangle}`

```
3791 \newcommand*{\@gls@entry@field}[2] {%
3792   \csname glo@glsdetoklabel{#1}@#2\endcsname
3793 }
```

\glsletentryfield

`\glsletentryfield{\langle cs \rangle}{\langle label \rangle}{\langle field \rangle}`

```

3794 \newcommand*\glsletentryfield}[3]{%
3795   \letcs{#1}{glo@\glsdetoklabel{#2}@#3}%
3796 }

```

`\@Gls@entry@field` Generic first letter uppercase version.

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

```

3797 \newcommand*\@Gls@entry@field}[2]{%
3798   \letcs\@glo@text{glo@\glsdetoklabel{#1}@#2}%
3799   \ifdef\@glo@text
3800     {%
3801       \xmakefirstuc\@glo@text}%
3802     }%
3803     {%
3804       \PackageError{glossaries}{Either glossary entry
3805         '\glsdetoklabel{#1}' doesn't exist or the field '#2'
3806         doesn't exist}{Check you have correctly spelt the entry
3807         label and the field name}%
3808     }%
3809 }

```

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`

```
3810 \newcommand*\glsentryname}[1]{\@Gls@entry@field{#1}{name}}
```

`\Glsentryname`

```

3811 \newrobustcmd*\Glsentryname}[1]{%
3812   \@Gls@entryname{#1}%
3813 }

```

`\@Gls@entryname` This is a workaround in the event that the user defies the warning in the manual about not using `\Glsname` or `\Glsentryname` with acronyms. First the default behaviour:

```

3814 \newcommand*\@Gls@entryname}[1]{%
3815   \@Gls@entry@field{#1}{name}%
3816 }

```

`\@Gls@acrentryname` Now the behaviour when `\setacronymstyle` is used:

```

3817 \newcommand*\@Gls@acrentryname}[1]{%
3818   \ifglshaslong{#1}%
3819     {%
3820       \letcs\@glo@text{glo@\glsdetoklabel{#1}@name}%
3821       \expandafter\@gls@getbody\@glo@text{}\@nil

```

```

3822 \expandafter\ifx\@gls@body\glsentrylong\relax
3823 \expandafter\Glsentrylong\@gls@rest
3824 \else
3825 \expandafter\ifx\@gls@body\glsentryshort\relax
3826 \expandafter\Glsentryshort\@gls@rest
3827 \else
3828 \expandafter\ifx\@gls@body\acronymfont\relax

```

Temporarily make `\glsentryshort` behave like `\Glsentryshort`. (This is on the assumption that the argument of `\acronymfont` is `\glsentryshort{<label>}`, as that's the behaviour of the predefined acronym styles.) This is scoped to localise the effect of the assignment.

```

3829 {%
3830 \let\glsentryshort\Glsentryshort
3831 \@glo@text
3832 }%
3833 \else
3834 \xmakefirstuc{\@glo@text}%
3835 \fi
3836 \fi
3837 \fi
3838 }%
3839 {%

```

Not an acronym

```

3840 \@Gls@entry@field{#1}{name}%
3841 }%
3842 }

```

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`

```

3843 \newcommand*{\glsentrydesc}[1]{\@gls@entry@field{#1}{desc}}

```

`\Glsentrydesc`

```

3844 \newrobustcmd*{\Glsentrydesc}[1]{%
3845 \@Gls@entry@field{#1}{desc}%
3846 }

```

Plural form:

`\glsentrydescplural`

```

3847 \newcommand*{\glsentrydescplural}[1]{%
3848 \@gls@entry@field{#1}{descplural}%
3849 }

```

\Glsentrydescplural

```
3850 \newrobustcmd*{\Glsentrydescplural}[1]{%  
3851   \@Gls@entry@field{#1}{descplural}%  
3852 }
```

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

```
3853 \newcommand*{\glsentrytext}[1]{\@Gls@entry@field{#1}{text}}
```

\Glsentrytext

```
3854 \newrobustcmd*{\Glsentrytext}[1]{%  
3855   \@Gls@entry@field{#1}{text}%  
3856 }
```

Get the plural form:

\glsentryplural

```
3857 \newcommand*{\glsentryplural}[1]{%  
3858   \@Gls@entry@field{#1}{plural}%  
3859 }
```

\Glsentryplural

```
3860 \newrobustcmd*{\Glsentryplural}[1]{%  
3861   \@Gls@entry@field{#1}{plural}%  
3862 }
```

Get the symbol associated with this entry. The argument is the label associated with the entry.

\glsentrysymbol

```
3863 \newcommand*{\glsentrysymbol}[1]{%  
3864   \@Gls@entry@field{#1}{symbol}%  
3865 }
```

\Glsentrysymbol

```
3866 \newrobustcmd*{\Glsentrysymbol}[1]{%  
3867   \@Gls@entry@field{#1}{symbol}%  
3868 }
```

Plural form:

\glsentrysymbolplural

```
3869 \newcommand*{\glsentrysymbolplural}[1]{%  
3870   \@Gls@entry@field{#1}{symbolplural}%  
3871 }
```

lentrysymbolplural

```
3872 \newrobustcmd*{\Glsentrysymbolplural}[1]{%
3873   \@Gls@entry@field{#1}{symbolplural}%
3874 }
```

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

```
3875 \newcommand*{\glsentryfirst}[1]{%
3876   \@Gls@entry@field{#1}{first}%
3877 }
```

\Glsentryfirst

```
3878 \newrobustcmd*{\Glsentryfirst}[1]{%
3879   \@Gls@entry@field{#1}{first}%
3880 }
```

Get the plural form (as specified by the firstplural key when the entry was defined).

glsentryfirstplural

```
3881 \newcommand*{\glsentryfirstplural}[1]{%
3882   \@Gls@entry@field{#1}{firstpl}%
3883 }
```

Glsentryfirstplural

```
3884 \newrobustcmd*{\Glsentryfirstplural}[1]{%
3885   \@Gls@entry@field{#1}{firstpl}%
3886 }
```

Display the glossary type with which this entry is associated (as specified by the type key used when the entry was defined)

\glsentrytype

```
3887 \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

```
3888 \newcommand*{\glsentrysort}[1]{%
3889   \@Gls@entry@field{#1}{sort}%
3890 }
```

\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.

```
3891 \newcommand*{\glsentryuseri}[1]{%
3892   \@Gls@entry@field{#1}{useri}%
3893 }
```



```

\Glsentryuseri
3894 \newrobustcmd*{\Glsentryuseri}[1]{%
3895   \@Gls@entry@field{#1}{useri}%
3896 }

\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.
3897 \newcommand*{\glsentryuserii}[1]{%
3898   \@Gls@entry@field{#1}{userii}%
3899 }

\Glsentryuserii
3900 \newrobustcmd*{\Glsentryuserii}[1]{%
3901   \@Gls@entry@field{#1}{userii}%
3902 }

\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.
3903 \newcommand*{\glsentryuseriii}[1]{%
3904   \@Gls@entry@field{#1}{useriii}%
3905 }

\Glsentryuseriii
3906 \newrobustcmd*{\Glsentryuseriii}[1]{%
3907   \@Gls@entry@field{#1}{useriii}%
3908 }

\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.
3909 \newcommand*{\glsentryuseriv}[1]{%
3910   \@Gls@entry@field{#1}{useriv}%
3911 }

\Glsentryuseriv
3912 \newrobustcmd*{\Glsentryuseriv}[1]{%
3913   \@Gls@entry@field{#1}{useriv}%
3914 }

\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.
3915 \newcommand*{\glsentryuserv}[1]{%
3916   \@Gls@entry@field{#1}{userv}%
3917 }

\Glsentryuserv
3918 \newrobustcmd*{\Glsentryuserv}[1]{%
3919   \@Gls@entry@field{#1}{userv}%
3920 }

```

`\glentryuservi` Get the sixth user key (as specified by the user6 when the entry was defined).
The argument is the label associated with the entry.

```
3921 \newcommand*{\glentryuservi}[1]{%  
3922   \@gls@entry@field{#1}{uservi}%  
3923 }
```

`\Glsentryuservi`

```
3924 \newrobustcmd*{\Glsentryuservi}[1]{%  
3925   \@Gls@entry@field{#1}{uservi}%  
3926 }
```

`\glentryshort` Get the short key (as specified by the short the entry was defined). The argument is the label associated with the entry.

```
3927 \newcommand*{\glentryshort}[1]{\@gls@entry@field{#1}{short}}
```

`\Glsentryshort`

```
3928 \newrobustcmd*{\Glsentryshort}[1]{%  
3929   \@Gls@entry@field{#1}{short}%  
3930 }
```

`\glentryshortpl` Get the short plural key (as specified by the shortplural the entry was defined).
The argument is the label associated with the entry.

```
3931 \newcommand*{\glentryshortpl}[1]{\@gls@entry@field{#1}{shortpl}}
```

`\Glsentryshortpl`

```
3932 \newrobustcmd*{\Glsentryshortpl}[1]{%  
3933   \@Gls@entry@field{#1}{shortpl}%  
3934 }
```

`\glentrylong` Get the long key (as specified by the long the entry was defined). The argument is the label associated with the entry.

```
3935 \newcommand*{\glentrylong}[1]{\@gls@entry@field{#1}{long}}
```

`\Glsentrylong`

```
3936 \newrobustcmd*{\Glsentrylong}[1]{%  
3937   \@Gls@entry@field{#1}{long}%  
3938 }
```

`\glentrylongpl` Get the long plural key (as specified by the longplural the entry was defined).
The argument is the label associated with the entry.

```
3939 \newcommand*{\glentrylongpl}[1]{\@gls@entry@field{#1}{longpl}}
```

`\Glsentrylongpl`

```
3940 \newrobustcmd*{\Glsentrylongpl}[1]{%  
3941   \@Gls@entry@field{#1}{longpl}%  
3942 }
```

Short cut macros to access full form:

`\glsentryfull`

```
3943 \newcommand*\glsentryfull}[1]{%
3944   \acrfullformat{\glsentrylong{#1}}{\acronymfont{\glsentryshort{#1}}}%
3945 }
```

`\Glsentryfull`

```
3946 \newrobustcmd*\Glsentryfull}[1]{%
3947   \acrfullformat{\Glsentrylong{#1}}{\acronymfont{\glsentryshort{#1}}}%
3948 }
```

`\glsentryfullpl`

```
3949 \newcommand*\glsentryfullpl}[1]{%
3950   \acrfullformat{\glsentrylongpl{#1}}{\acronymfont{\glsentryshortpl{#1}}}%
3951 }
```

`\Glsentryfullpl`

```
3952 \newrobustcmd*\Glsentryfullpl}[1]{%
3953   \acrfullformat{\Glsentrylongpl{#1}}{\acronymfont{\glsentryshortpl{#1}}}%
3954 }
```

`\glsentrynumberlist` Displays the number list as is.

```
3955 \newcommand*\glsentrynumberlist}[1]{%
3956   \glsdoifexists{#1}%
3957   {%
3958     \@gls@entry@field{#1}{numberlist}%
3959   }%
3960 }
```

`\glsdisplaynumberlist` Formats the number list for the given entry label. Doesn't work with hyperref.

```
3961 \@ifpackageloaded{hyperref} {%
3962   \newcommand*\glsdisplaynumberlist}[1]{%
3963     \GlossariesWarning
3964     {%
3965       \string\glsdisplaynumberlist\space
3966       doesn't work with hyperref.^^JUsing
3967       \string\glsentrynumberlist\space instead%
3968     }%
3969     \glsentrynumberlist{#1}%
3970   }%
3971 }%
3972 {%
3973   \newcommand*\glsdisplaynumberlist}[1]{%
3974     \glsdoifexists{#1}%
3975     {%
3976       \bgroup
```

```

3977 \edef\@glo@label{\glsdetoklabel{#1}}%
3978 \let\@org@glsglnumberformat\glsglnumberformat
3979 \def\glsglnumberformat##1{##1}%
3980 \protected@edef\the@numberlist{%
3981   \csname glo@\@glo@label @numberlist\endcsname}%
3982 \def\@glsgl@numlist@sep{}%
3983 \def\@glsgl@numlist@nextsep{}%
3984 \def\@glsgl@numlist@lastsep{}%
3985 \def\@glsgl@thislist{}%
3986 \def\@glsgl@donext@def{}%
3987 \renewcommand\do[1]{%
3988   \protected@edef\@glsgl@thislist{%
3989     \@glsgl@thislist
3990     \noexpand\@glsgl@numlist@sep
3991     ##1%
3992   }%
3993   \let\@glsgl@numlist@sep\@glsgl@numlist@nextsep
3994   \def\@glsgl@numlist@nextsep{\glsgl@numlist@sep}%
3995   \@glsgl@donext@def
3996   \def\@glsgl@donext@def{%
3997     \def\@glsgl@numlist@lastsep{\glsgl@numlist@lastsep}%
3998   }%
3999 }%
4000 \expandafter \glsgl@numlist@parser \expandafter{\the@numberlist}%
4001 \let\@glsgl@numlist@sep\@glsgl@numlist@lastsep
4002 \@glsgl@thislist
4003 \egroup
4004 }%
4005 }
4006 }

```

\glsgl@numlist@sep

```
4007 \newcommand*{\glsgl@numlist@sep}{, }
```

\glsgl@numlist@lastsep

```
4008 \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.

```

4009 \newcommand*{\glshyperlink}[2][\glsgl@entrytext{\@glo@label}]{%
4010   \def\@glo@label{#2}%
4011   \@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`:

```

4012 \define@key{glossadd}{counter}{\def\@gls@counter{#1}}
4013 \define@key{glossadd}{format}{\def\@glsnumberformat{#1}}

```

This key is only used by `\glsaddall`:

```

4014 \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`

```

4015 \newrobustcmd*{\glsadd}[2][]{%
4016   \glsdoifexists{#2}%
4017   {%
4018     \def\@glsnumberformat{glsnumberformat}%
4019     \edef\@gls@counter{\csname glo@glsetoklabel{#2}@counter\endcsname}%
4020     \setkeys{glossadd}{#1}%
4021     \@gls@saveentrycounter
4022     \@do@wrglossary{#2}%
4023   }%
4024 }

```

Store the entry's counter in `\theglsentrycounter`

`\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`

```

4025 \newrobustcmd*{\glsaddall}[1][]{%
4026   \edef\@glo@type{\@glo@types}%
4027   \setkeys{glossadd}{#1}%
4028   \forallglsentries[\@glo@type]{\@glo@entry}{%
4029     \glsadd[#1]{\@glo@entry}%
4030   }%
4031 }

```

`\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.

```

4032 \newrobustcmd*{\glsaddallunused}[1][\@glo@types]{%
4033 \forallglsentries[#1]{\@glo@entry}%
4034 {%
4035 \ifglsused{\@glo@entry}{\glsadd[format=@gobble]{\@glo@entry}}%
4036 }%
4037 }

```

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.

```
4038 \edef\glsopenbrace{\expandafter\@gobble\string\{}
```

`\glsclosebrace` Define `\glsclosebrace` to make it easier to write an opening brace to a file.

```
4039 \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.

```
4040 \edef\glsquote#1{\string"#1\string"}
```

`\glspercentchar` Define `\glspercentchar` to make it easier to write a percent character to a file.

```
4041 \edef\glspercentchar{\expandafter\@gobble\string\%}
```

`\glstildechar` Define `\glstildechar` to make it easier to write a tilde character to a file.

```
4042 \edef\glstildechar{\string~}
```

`\@glsfirstletter` Define the first letter to come after the digits 0,...,9. Only required for `xindy`.

```

4043 \ifglsxindy
4044 \newcommand*{\@glsfirstletter}{A}
4045 \fi

```

stLetterAfterDigits Sets the first letter to come after the digits 0,...,9.

```
4046 \ifglxindy
4047   \newcommand*{\GlsSetXdyFirstLetterAfterDigits}[1]{%
4048     \renewcommand*{\@glstfirstletter}{#1}}
4049 \else
4050   \newcommand*{\GlsSetXdyFirstLetterAfterDigits}[1]{%
4051     \glsnxindywarning\GlsSetXdyFirstLetterAfterDigits}
4052 \fi
```

\@glsminrange Define the minimum number of successive location references to merge into a range.

```
4053 \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.

```
4054 \ifglxindy
4055   \newcommand*{\GlsSetXdyMinRangeLength}[1]{%
4056     \renewcommand*{\@glsminrange}{#1}}
4057 \else
4058   \newcommand*{\GlsSetXdyMinRangeLength}[1]{%
4059     \glsnxindywarning\GlsSetXdyMinRangeLength}
4060 \fi
```

\writeist

```
4061 \ifglxindy
```

Code to use if xindy is required.

```
4062 \def\writeist{%
```

Define write register if not already defined

```
4063   \ifundef{\glswrite}{\newwrite\glswrite}{}%
```

Update attributes list

```
4064   \@glx@addpredefinedattributes
```

Open the file.

```
4065   \openout\glswrite=\istfilename
```

Write header comment at the start of the file

```
4066   \write\glswrite{;; xindy style file created by the glossaries
4067     package}%
```

```
4068   \write\glswrite{;; for document '\jobname' on
```

```
4069     \the\year-\the\month-\the\day}%
```

Specify the required styles

```
4070   \write\glswrite{^^J; required styles^^J}
4071   \@for\@xdystyle:=\@xdyrequiredstyles\do{%
4072     \ifx\@xdystyle\@empty
4073     \else
```

```

4074         \protected@write\glswrite{}\{(require
4075         \string"\@xdystyle.xdy\string")}\}%
4076     \fi
4077 }%

```

List the allowed attributes (possible values used by the format key)

```

4078     \write\glswrite{^^J%
4079         ; list of allowed attributes (number formats)^^J}%
4080     \write\glswrite{(define-attributes ((\@xdyattributes)))}%

```

Define any additional alphabets

```

4081     \write\glswrite{^^J; user defined alphabets^^J}%
4082     \write\glswrite{\@xdyuseralphabets}%

```

Define location classes.

```

4083     \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.

```

4084     \@for\@gls@classI:=\@gls@xdy@locationlist\do{%

```

Case were $\langle Hprefix \rangle$ is empty:

```

4085         \protected@write\glswrite{}\{(define-location-class
4086         \string"\@gls@classI\string"^^J\space\space\space
4087         (
4088             :sep "{{"
4089             \csname @gls@xdy@Lclass@\@gls@classI\endcsname\space
4090             :sep "}"
4091         )
4092         ^^J\space\space\space
4093         :min-range-length \@glsminrange^^J%
4094     )
4095 }%

```

Nested iteration over all classes:

```

4096     {%
4097         \@for\@gls@classII:=\@gls@xdy@locationlist\do{%
4098             \protected@write\glswrite{}\{(define-location-class
4099             \string"\@gls@classII-\@gls@classI\string"
4100             ^^J\space\space\space
4101             (
4102                 :sep "{"
4103                 \csname @gls@xdy@Lclass@\@gls@classII\endcsname\space
4104                 :sep "}"
4105                 \csname @gls@xdy@Lclass@\@gls@classI\endcsname\space
4106                 :sep "}"
4107             )
4108             ^^J\space\space\space
4109             :min-range-length \@glsminrange^^J%
4110         )
4111     }%
4112 }%

```



```

4113     }%
4114 }%

```

User defined location classes (needs checking for new location format).

```

4115 \write\glswrite{^^J; user defined location classes}%
4116 \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.)

```

4117 \write\glswrite{^^J; define cross-reference class^^J}%
4118 \write\glswrite{(define-crossref-class \string"see\string"
4119 :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.)

```

4120 \write\glswrite{(markup-crossref-list
4121 :class \string"see\string"^^J\space\space\space
4122 :open \string"\string\glsseeformat\string"
4123 :close \string"{}\string")}%

```

List the order to sort the classes.

```

4124 \write\glswrite{^^J; define the order of the location classes}%
4125 \write\glswrite{(define-location-class-order
4126 (\@xdylocationclassorder))}%

```

Specify what to write to the start and end of the glossary file.

```

4127 \write\glswrite{^^J; define the glossary markup^^J}%

4128 \write\glswrite{(markup-index^^J\space\space\space
4129 :open \string"\string
4130 \glossarysection[\string\glossarytoctitle]{\string
4131 \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`)

```

4132 \@for\@this@ctr:=\@xdycounters\do{%
4133   {%
4134     \@for\@this@attr:=\@xdyattributelist\do{%
4135       \protected@write\glswrite{{\string\providecommand*%
4136         \expandafter\string
4137         \csname glsX\@this@ctr X\@this@attr\endcsname[2]%
4138         {%
4139           \string\setentrycounter
4140             [\expandafter\@gobble\string\#1]{\@this@ctr}%
4141           \expandafter\string
4142           \csname\@this@attr\endcsname
4143             {\expandafter\@gobble\string\#2}%
4144         }}%

```

```

4145         }%
4146     }%
4147 }%
4148 }%

```

Add the end part of the open tag and the rest of the markup-index information:

```

4149 \write\glswrite{%
4150     \string\begin
4151     {theglossary}\string\glossaryheader\glstildechar n\string" ^^J\space
4152     \space\space:close \string"\glpercentchar\glstildechar n\string
4153     \end{theglossary}\string\glossarypostamble
4154     \glstildechar n\string" ^^J\space\space\space
4155     :tree)}}%

```

Specify what to put between letter groups

```

4156 \write\glswrite{(markup-letter-group-list
4157     :sep \string"\string\glsgroupskip\glstildechar n\string"))}%

```

Specify what to put between entries

```

4158 \write\glswrite{(markup-indexentry
4159     :open \string"\string\relax \string\glsresetentrylist
4160     \glstildechar n\string"))}%

```

Specify how to format entries

```

4161 \write\glswrite{(markup-locclass-list :open
4162     \string"\glsoopenbrace\string\glossaryentrynumbers
4163     \glsoopenbrace\string\relax\space \string"^^J\space\space\space
4164     :sep \string", \string"
4165     :close \string"\glsclosebrace\glsclosebrace\string"))}%

```

Specify how to separate location numbers

```

4166 \write\glswrite{(markup-locref-list
4167     :sep \string"\string\delimN\space\string"))}%

```

Specify how to indicate location ranges

```

4168 \write\glswrite{(markup-range
4169     :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.

```

4170 \@onelevel@sanitize\gls@suffixF
4171 \@onelevel@sanitize\gls@suffixFF
4172 \ifx\gls@suffixF\@empty
4173 \else
4174     \write\glswrite{(markup-range
4175         :close "\gls@suffixF" :length 1 :ignore-end)}}%
4176 \fi
4177 \ifx\gls@suffixFF\@empty
4178 \else
4179     \write\glswrite{(markup-range
4180         :close "\gls@suffixFF" :length 2 :ignore-end)}}%
4181 \fi

```

Specify how to format locations.

```
4182 \write\glswrite{^^J; define format to use for locations^^J}%
4183 \write\glswrite{\@xdylocref}%
```

Specify how to separate letter groups.

```
4184 \write\glswrite{^^J; define letter group list format^^J}%
4185 \write\glswrite{(markup-letter-group-list
4186 :sep \string\string\glsgroupskip\glstildechar n\string)}}%
```

Define letter group headings.

```
4187 \write\glswrite{^^J; letter group headings^^J}%
4188 \write\glswrite{(markup-letter-group
4189 :open-head \string\string\glsgroupheading
4190 \glsoopenbrace\string^^J\space\space\space
4191 :close-head \string\glsclosebrace\string)}}%
```

Define additional letter groups.

```
4192 \write\glswrite{^^J; additional letter groups^^J}%
4193 \write\glswrite{\@xdylettergroups}%
```

Define additional sort rules

```
4194 \write\glswrite{^^J; additional sort rules^^J}
4195 \write\glswrite{\@xdysortrules}%
```

Close the style file

```
4196 \closeout\glswrite
```

Suppress any further calls.

```
4197 \let\writeist\relax
4198 }
4199 \else
```

Code to use if makeindex is required.

```
4200 \edef\@gls@actualchar{\string?}
4201 \edef\@gls@encapchar{\string|}
4202 \edef\@gls@levelchar{\string!}
4203 \edef\@gls@quotechar{\string"}
4204 \def\writeist{\relax
4205 \ifundef\glswrite{\newwrite\glswrite}{\relax
4206 \openout\glswrite=\istfilename
4207 \write\glswrite{\glspersentchar\space makeindex style file
4208 created by the glossaries package}
4209 \write\glswrite{\glspersentchar\space for document
4210 '\jobname' on \the\year-\the\month-\the\day}
4211 \write\glswrite{actual '\@gls@actualchar'}
4212 \write\glswrite{encap '\@gls@encapchar'}
4213 \write\glswrite{level '\@gls@levelchar'}
4214 \write\glswrite{quote '\@gls@quotechar'}
4215 \write\glswrite{keyword \string\string\glossaryentry\string"}
4216 \write\glswrite{preamble \string\string\glossarysection[\string
4217 \glossarytoctitle]{\string\glossarytitle}\string
4218 \glossarypreamble\string\n\string\begin{theglossary}\string
```

```

4219     \glossaryheader\string\n\string"}
4220 \write\glswrite{postamble \string"\string%\string\n\string
4221     \end{theglossary}\string\glossarypostamble\string\n
4222     \string"}
4223 \write\glswrite{group_skip \string"\string\glsgroupskip\string\n
4224     \string"}
4225 \write\glswrite{item_0 \string"\string%\string\n\string"}
4226 \write\glswrite{item_1 \string"\string%\string\n\string"}
4227 \write\glswrite{item_2 \string"\string%\string\n\string"}
4228 \write\glswrite{item_01 \string"\string%\string\n\string"}
4229 \write\glswrite{item_x1
4230     \string"\string\relax \string\glresetentrylist\string\n
4231     \string"}
4232 \write\glswrite{item_12 \string"\string%\string\n\string"}
4233 \write\glswrite{item_x2
4234     \string"\string\relax \string\glresetentrylist\string\n
4235     \string"}

4236 \write\glswrite{delim_0 \string"\string\{\string
4237     \glossaryentrynumbers\string\{\string\relax \string"}
4238 \write\glswrite{delim_1 \string"\string\{\string
4239     \glossaryentrynumbers\string\{\string\relax \string"}
4240 \write\glswrite{delim_2 \string"\string\{\string
4241     \glossaryentrynumbers\string\{\string\relax \string"}
4242 \write\glswrite{delim_t \string"\string\}\string\}\string"}
4243 \write\glswrite{delim_n \string"\string\delimN \string"}
4244 \write\glswrite{delim_r \string"\string\delimR \string"}
4245 \write\glswrite{headings_flag 1}
4246 \write\glswrite{heading_prefix
4247     \string"\string\glsgroupheading\string\{\string"}
4248 \write\glswrite{heading_suffix
4249     \string"\string\}\string\relax
4250     \string\glresetentrylist \string"}
4251 \write\glswrite{symhead_positive \string"glssymbols\string"}
4252 \write\glswrite{numhead_positive \string"glsnnumbers\string"}
4253 \write\glswrite{page_compositor \string"glscpositor\string"}
4254 \@gls@escbsdq\gls@suffixF
4255 \@gls@escbsdq\gls@suffixFF
4256 \ifx\gls@suffixF\@empty
4257 \else
4258     \write\glswrite{suffix_2p \string"\gls@suffixF\string"}
4259 \fi
4260 \ifx\gls@suffixFF\@empty
4261 \else
4262     \write\glswrite{suffix_3p \string"\gls@suffixFF\string"}
4263 \fi
4264 \closeout\glswrite
4265 \let\writeist\relax
4266 }
4267 \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`

```
4268 \newcommand{\noist}{%
```

Update attributes list

```
4269 \@gls@addpredefinedattributes
```

```
4270 \let\writeist\relax
```

```
4271 }
```

`\@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`

```
4272 \newcommand*{\@makeglossary}[1]{%
```

```
4273 \ifglossaryexists{#1}%
```

```
4274 {%
```

Only create a new write if `savewrites=false` otherwise create a token to collect the information.

```
4275 \ifglssavewrites
```

```
4276 \expandafter\newtoks\csname glo@#1@filetok\endcsname
```

```
4277 \else
```

```
4278 \expandafter\newwrite\csname glo@#1@file\endcsname
```

```
4279 \expandafter\@glsopenfile\csname glo@#1@file\endcsname{#1}%
```

```
4280 \fi
```

```
4281 \@gls@renewglossary
```

```
4282 \writeist
```

```
4283 }%
```

```
4284 {%
```

```
4285 \PackageError{glossaries}%
```

```
4286 {Glossary type ‘#1’ not defined}%
```

```
4287 {New glossaries must be defined before using \string\makeglossary}%
```

```
4288 }%
```

```
4289 }
```

`\@glsopenfile` Open write file associated with the given glossary.

```
4290 \newcommand*{\@glsopenfile}[2]{%
```

```
4291 \immediate\openout#1=\jobname.\csname @glotype@#2@out\endcsname
```

```
4292 \PackageInfo{glossaries}{Writing glossary file
```

```

4293     \jobname.\csname @glotype@#2@out\endcsname}%
4294 }

\@closegls

4295 \newcommand*{\@closegls}[1]{%
4296   \closeout\csname glo@#1@file\endcsname
4297 }
4298 %   \end{macrocode}
4299 %\end{macro}
4300 %
4301 %\begin{macro}{\@gls@automake}
4302 %\changes{4.08}{2014-07-30}{new}
4303 %   \begin{macrocode}
4304 \ifglxindy
4305   \newcommand*{\@gls@automake}[1]{%
4306     \ifglossaryexists{#1}
4307     {%
4308       \@closegls{#1}%
4309       \ifdefstring{\glsorder}{letter}%
4310         {\def\@gls@order{-M ord/letorder }}%
4311         {\let\@gls@order\@empty}%
4312       \ifcsundef{@xdy@#1@language}%
4313         {\let\@gls@langmod\@xdy@main@language}%
4314         {\letcs\@gls@langmod{@xdy@#1@language}}%
4315       \edef\@gls@dothiswrite{\noexpand\write18{xindy
4316         -I xindy
4317         \@gls@order
4318         -L \@gls@langmod\space
4319         -M \@gls@istfilebase\space
4320         -C \@gls@codepage\space
4321         -t \jobname.\csuse{@glotype@#1@log}
4322         -o \jobname.\csuse{@glotype@#1@in}
4323         \jobname.\csuse{@glotype@#1@out}}}%
4324     }%
4325     \@gls@dothiswrite
4326   }%
4327   {%
4328     \GlossariesWarning{Can't make glossary '#1', it doesn't exist}%
4329   }%
4330 }
4331 \else
4332 \newcommand*{\@gls@automake}[1]{%
4333   \ifglossaryexists{#1}
4334   {%
4335     \@closegls{#1}%
4336     \ifdefstring{\glsorder}{letter}%
4337       {\def\@gls@order{-l }}%
4338       {\let\@gls@order\@empty}%
4339     \edef\@gls@dothiswrite{\noexpand\write18{makeindex \@gls@order

```

```

4340      -s \istfilename\space
4341      -t \jobname.\csuse{@glotype@#1@log}
4342      -o \jobname.\csuse{@glotype@#1@in}
4343      \jobname.\csuse{@glotype@#1@out}}}%
4344  }%
4345  \@gls@dothiswrite
4346  }%
4347  {%
4348  \GlossariesWarning{Can't make glossary '#1', it doesn't exist}%
4349  }%
4350 }
4351 \fi

```

`\nomakeglossaries` Issue warning that `\makeglossaries` hasn't been used.

```

4352 \newcommand*{\@warn@nomakeglossaries}{}

```

Only use this if warning if `\printglossary` has been used without `\makeglossaries`

```

4353 \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`

```

4354 \newcommand*{\makeglossaries}{%

```

Define the write used for style file also used for all other output files if `savewrites=true`.

```

4355 \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.

```

4356 \protected@write\@auxout{}{\string\providecommand\string\@glsorder[1]{} }
4357 \protected@write\@auxout{}{\string\providecommand\string\@istfilename[1]{} }

```

Write the name of the style file to the aux file (needed by `makeglossaries`)

```

4358 \protected@write\@auxout{}{\string\@istfilename{\istfilename}}%
4359 \protected@write\@auxout{}{\string\@glsorder{\glsorder}}

```

Iterate through each glossary type and activate it.

```

4360 \@for\@glo@type:=\@glo@types\do{%
4361   \ifthenelse{\equal{\@glo@type}{}}{}{}%
4362   \@makeglossary{\@glo@type}}%
4363 }%

```

New glossaries must be created before `\makeglossaries` so disable `\newglossary`.

```

4364 \renewcommand*\newglossary[4] []{%
4365 \PackageError{glossaries}{New glossaries
4366 must be created before \string\makeglossaries}{You need

```

```

4367 to move \string\makeglossaries\space after all your
4368 \string\newglossary\space commands}}}%

```

Any subsequence instances of this command should have no effect

```

4369 \let\@makeglossary\relax
4370 \let\makeglossary\relax
4371 \let\makeglossaries\relax

```

Disable all commands that have no effect after \makeglossaries

```

4372 \@disable@onlypremakeg

```

Allow see key:

```

4373 \let\gls@checkseeallowed\relax

```

Suppress warning about no \makeglossaries

```

4374 \let\warn@nomakeglossaries\relax

```

Activate warning about missing \printglossary

```

4375 \def\warn@noprintglossary{%
4376   \GlossariesWarningNoLine{No \string\printglossary\space
4377     or \string\printglossaries\space
4378     found.^^J(Remove \string\makeglossaries\space if you don't want
4379     any glossaries.)^^JThis document will not have a glossary}%
4380 }%

```

Declare list parser for \glsdisplaynumberlist

```

4381 \ifglssavenumberlist
4382   \edef\@gls@dodolistparser{\noexpand\DeclareListParser
4383     {\noexpand\glsnumlistparser}{\delimN}}}%
4384   \@gls@dodolistparser
4385 \fi

```

Prevent user from also using \makenoidxglossaries

```

4386 \let\makenoidxglossaries\@no@makeglossaries

```

Prohibit sort key in printgloss family:

```

4387 \renewcommand*{\@printgloss@setsort}{%
4388   \let\@glo@assign@sortkey\@glo@no@assign@sortkey
4389 }%

```

Check the automake setting:

```

4390 \ifglssautomake
4391   \renewcommand*{\@gls@doautomake}{%
4392     \@for\@gls@type:=\@glo@types\do{%
4393       \ifdefempty{\@gls@type}{}%
4394       {\@gls@automake{\@gls@type}}%
4395     }%
4396   }%
4397 \fi
4398 }

```

Must occur in the preamble:

```

4399 \@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`

```
4400 \let\makeglossary\makeglossaries
```

If `\makeglossaries` hasn't been used, issue a warning. Also issue a warning if neither `\printglossaries` nor `\printglossary` have been used.

```
4401 \AtEndDocument{%
```

```
4402   \warn@nomakeglossaries
```

```
4403   \warn@noprintglossary
```

```
4404 }
```

`\makenoidxglossaries` Analogous to `\makeglossaries` this activates the commands needed for `\printnoidxglossary`

```
4405 \newcommand*\makenoidxglossaries{%
```

Redefine empty glossary warning:

```
4406   \renewcommand{\@gls@noref@warn}[1]{%
```

```
4407     \GlossariesWarning{Empty glossary for
```

```
4408     \string\printnoidxglossary[type={##1}].
```

```
4409     Rerun may be required (or you may have forgotten to use
```

```
4410     commands like \string\gls).}%
```

```
4411   }%
```

Don't escape `makeindex`/`xindy` characters

```
4412   \let\@gls@checkmkidxchars\@gobble
```

Write glossary information to aux instead of glossary files

```
4413   \let\@@do@wrglossary\gls@noidxglossary
```

Switch on group headings that use the character code:

```
4414   \let\@gls@getgrouptitle\@gls@noidx@getgrouptitle
```

Allow see key:

```
4415   \let\gls@checkseeallowed\relax
```

Redefine cross-referencing macro:

```
4416   \renewcommand{\@do@seeglossary}[2]{%
```

```
4417     \edef\@gls@label{\glsdetoklabel{##1}}%
```

```
4418     \protected@write\@auxout{}{%
```

```
4419       \string\@gls@reference
```

```
4420       {\csname glo@\@gls@label @type\endcsname}%
```

```
4421       {\@gls@label}%
```

```
4422       {%
```

```
4423       \string\glsseeformat##2}%
```

```
4424       }%
```

```
4425     }%
```

```
4426   }%
```

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.

```
4427 \AtBeginDocument
4428 {%
4429 \write\@auxout{\string\providecommand\string\@gls@reference[3]{}}%
4430 }%
```

Change warning about no glossares

```
4431 \def\warn@noprintglossary{%
4432 \GlossariesWarningNoLine{No \string\printnoidxglossary\space
4433 or \string\printnoidxglossaries ^^J
4434 found. (Remove \string\makenoidxglossaries\space if you
4435 don't want any glossaries.)^^JThis document will not have a glossary}%
4436 }%
```

Suppress warning about no \makeglossaries

```
4437 \let\warn@nomakeglossaries\relax
```

Prevent user from also using \makeglossaries

```
4438 \let\makeglossaries\@no@makeglossaries
```

Allow sort key in printgloss family:

```
4439 \renewcommand*{\@printgloss@setsort}{%
4440 \let\@glo@assign@sortkey\@glo@assign@sortkey
```

Initialise default sort order:

```
4441 \def\@glo@sorttype{\@glo@default@sorttype}%
4442 }%
```

All entries must be defined in the preamble:

```
4443 \renewcommand*\new@glossaryentry[2]{%
4444 \PackageError{glossaries}{Glossary entries must be
4445 defined in the preamble^^Jwhen you use
4446 \string\makenoidxglossaries}%
4447 {Either move your definitions to the preamble or use
4448 \string\makeglossaries}%
4449 }%
```

Redefine \glsentrynumberlist

```
4450 \renewcommand*{\glsentrynumberlist}[1]{%
4451 \letcs{\@gls@loclist}{glo@\glsdetoklabel{##1}@loclist}%
4452 \ifdef\@gls@loclist
4453 {%
4454 \glsnoidxloclist{\@gls@loclist}%
4455 }%
4456 {%
4457 \ifglsentryexists{##1}%
4458 {%
4459 \GlossariesWarning{Missing location list for ‘##1’. Either
4460 a rerun is required or you haven't referenced the entry.}%
4461 }%
```

```

4462     {%
4463         \PackageError{glossaries}{Glossary entry ‘##1’ has not been
4464             defined.}{}%
4465     }%
4466 }%
4467 }%

Redefine \glsdisplaynumberlist
4468 \renewcommand*{\glsdisplaynumberlist}[1]{%
4469     \letcs{\@gls@loclist}{glo@\glsdetoklabel{##1}@loclist}%
4470     \ifdef\@gls@loclist
4471     {%
4472         \def\@gls@noidxloclist@sep{%
4473             \def\@gls@noidxloclist@sep{%
4474                 \def\@gls@noidxloclist@sep{%
4475                     \glsnumlistsep
4476                 }%
4477             \def\@gls@noidxloclist@finalsep{\glsnumlistlastsep}%
4478         }%
4479     }%
4480     \def\@gls@noidxloclist@finalsep{}%
4481     \def\@gls@noidxloclist@prev{}%
4482     \forlistloop{\glsnoidxdisplayloclisthandler}{\@gls@loclist}%
4483     \@gls@noidxloclist@finalsep
4484     \@gls@noidxloclist@prev
4485 }%
4486 {%
4487     ??\ifglsentryexists{##1}%
4488     {%
4489         \GlossariesWarning{Missing location list for ‘##1’. Either
4490             a rerun is required or you haven’t referenced the entry.}%
4491     }%
4492     {%
4493         \PackageError{glossaries}{Glossary entry ‘##1’ has not been
4494             defined.}{}%
4495     }%
4496 }%
4497 }%

```

Provide a generic way of iterating through the number list:

```

4498 \renewcommand*{\glsnumberlistloop}[3]{%
4499     \letcs{\@gls@loclist}{glo@\glsdetoklabel{##1}@loclist}%
4500     \let\@gls@org@glsnoidxdisplayloc\glsnoidxdisplayloc
4501     \let\@gls@org@glsseeformat\glsseeformat
4502     \let\glsnoidxdisplayloc##2\relax
4503     \let\glsseeformat##3\relax
4504     \ifdef\@gls@loclist
4505     {%
4506         \forlistloop{\glsnoidxnumberlistloophandler}{\@gls@loclist}%
4507     }%

```

```

4508    {%
4509        \ifglsentryexists{##1}%
4510        {%
4511            \GlossariesWarning{Missing location list for ‘##1’. Either
4512                a rerun is required or you haven’t referenced the entry.}%
4513        }%
4514        {%
4515            \PackageError{glossaries}{Glossary entry ‘##1’ has not been
4516                defined.}{}%
4517        }%
4518    }%
4519    \let\glsnoidxdisplayloc\@gls@org\glsnoidxdisplayloc
4520    \let\glsseeformat\@gls@org\glsseeformat
4521 }%

```

Modify sanitize sort function

```

4522 \let\@gls@sanitizesort\@gls@noidx@sanitizesort
4523 \let\@gls@nosanitizesort\@gls@noidx@nosanitizesort
4524 \@gls@noidx@setsanitizesort
4525 }

```

Preamble-only command:

```

4526 \onlypreamble{\makenoidxglossaries}

```

`\glsnumberlistloop` `\glsnumberlistloop{<label>}{<handler>}`

```

4527 \newcommand*{\glsnumberlistloop}[2]{%
4528     \PackageError{glossaries}{\string\glsnumberlistloop\space
4529         only works with \string\makenoidxglossaries}{}%
4530 }

```

`numberlistloophandler` Handler macro for `\glsnumberlistloop`. (The argument should be in the form `\glsnoidxdisplayloc{<prefix>}{<counter>}{<format>}{<n>}`)

```

4531 \newcommand*{\glsnoidxnumberlistloophandler}[1]{%
4532     #1%
4533 }

```

`\@no@makeglossaries` Can’t use both `\makeglossaries` and `\makenoidxglossaries`

```

4534 \newcommand*{\@no@makeglossaries}{%
4535     \PackageError{glossaries}{You can’t use both
4536         \string\makeglossaries\space and \string\makenoidxglossaries}%
4537     {Either use one or other (or none) of those commands but not both
4538         together.}%
4539 }

```

`\@gls@noref@warn` Warning when no instances of `\@gls@reference` found.

```

4540 \newcommand{\@gls@noref@warn}[1]{%
4541     \GlossariesWarning{\string\makenoidxglossaries\space

```

```

4542   is required to make \string\printnoidxglossary[type={#1}] work}%
4543 }

```

`\gls@noidxglossary` Write the glossary information to the aux file:

```

4544 \newcommand*{\gls@noidxglossary}{%
4545   \protected@write\@auxout{}{%
4546     \string\gls@reference
4547     {\csname glo@\gls@label @type\endcsname}%
4548     {\gls@label}%
4549     {\string\glsnoidxdisplayloc
4550      {\gls@counterprefix}%
4551      {\gls@counter}%
4552      {\glsnumberformat}%
4553      {\glslocref}%
4554     }%
4555   }%
4556 }

```

1.13 Writing information to associated files

`\istfile` Deprecated.

```

4557 \def\istfile{\glswrite}

```

At the end of the document, the files should be created if `savewrites=true`.

```

4558 \AtEndDocument{%
4559   \glswritefiles
4560 }

```

`\@glswritefiles` Only write the files if `savewrites=true`

```

4561 \newcommand*{\@glswritefiles}{%

```

Iterate through all the glossaries

```

4562   \forallglossaries{\glo@type}{%

```

Check for empty glossaries (patch provided by Patrick Häcker)

```

4563     \ifcsundef{glo@\glo@type @filetok}%
4564     {%
4565       \def\gls@tmp{}%
4566     }%
4567     {%
4568       \edef\gls@tmp{\expandafter\the
4569         \csname glo@\glo@type @filetok\endcsname}%
4570     }%
4571     \ifx\gls@tmp\@empty
4572       \ifx\glo@type\glsdefaulttype
4573         \GlossariesWarningNoLine{Glossary '\glo@type' has no
4574           entries.^^JRemember to use package option 'nomain' if
4575 you
4576           don't want to^^Juse the main glossary}%

```

```

4577         \else
4578             \GlossariesWarningNoLine{Glossary ‘\@glo@type’ has no
4579                 entries}%
4580         \fi
4581     \else
4582         \@glsopenfile{\glswrite}{\@glo@type}%
4583         \immediate\write\glswrite{%
4584             \expandafter\the
4585             \csname glo@\@glo@type @filetok\endcsname}%
4586         \immediate\closeout\glswrite
4587     \fi
4588 }%
4589 }

```

As from v4.10, the `\glossary` command is no longer redefined and no longer used by the glossaries package. Since the user isn't expected to use this command (as glossaries takes care of the particular format required for `makeindex/xindy`) there's no need for a user level command. Using a custom internal command prevents any conflict with other packages (and with the `\mark` mechanism).

The associated number should be stored in `\theglsentrycounter` before using `\gls@glossary`.

`\gls@glossary`

```

4590 \newcommand*{\gls@glossary}[1]{%
4591     \@gls@glossary{#1}%
4592 }

```

`\@gls@glossary` (In v4.10, `\@glossary` was redefined to `\@gls@glossary` to avoid conflict with other packages.) 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.)

`\gls@glossary`

```

4593 \newcommand*{\@gls@glossary}[1]{\index}

```

This is a convenience command to set `\@gls@glossary`. It's used by `\@makeglossary` and then redefined to do nothing, as it only needs to be done once.

`\@gls@renewglossary`

```

4594 \newcommand{\@gls@renewglossary}{%
4595     \gdef\@gls@glossary##1{\@bsphack\begingroup\@wrglossary{##1}}%
4596     \let\@gls@renewglossary\@empty
4597 }

```

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

```

4598 \renewcommand*{\@wrglossary}[2]{%
4599   \ifglssavewrites
4600     \protected@edef\@gls@tmp{\the\csname glo@#1@filetok\endcsname#2}%
4601     \expandafter\global\expandafter\csname glo@#1@filetok\endcsname
4602       \expandafter{\@gls@tmp^^J}%
4603   \else

4604     \ifcsdef{glo@#1@file}%
4605     {%
4606       \expandafter\protected@write\csname glo@#1@file\endcsname{%
4607         \gls@disablepagerefexpansion}{#2}%
4608     }%
4609     {%
4610       \ifignoredglossary{#1}{}%
4611       {%
4612         \GlossariesWarning{No file defined for glossary ‘#1’}%
4613       }%
4614     }%
4615   \fi
4616 \endgroup\@esphack
4617 }

```

\@do@wrglossary

```

4618 \newcommand*{\@do@wrglossary}[1]{%
4619   \ifglsindexonlyfirst
4620     \ifglsused{#1}{\@do@wrglossary{#1}}%
4621   \else
4622     \@do@wrglossary{#1}%
4623   \fi
4624 }

```

@protected@pagefmts List of page formats to be protected against expansion.

```

4625 \newcommand{\gls@protected@pagefmts}{%
4626   \gls@numberpage,\gls@alphpage,\gls@Alphpage,\gls@romanpage,\gls@Romanpage%
4627 }

```

blepagerefexpansion

```

4628 \newcommand*{\gls@disablepagerefexpansion}{%
4629   \@for\@gls@this:=\gls@protected@pagefmts\do
4630   {%
4631     \expandafter\let\@gls@this\relax
4632   }%
4633 }

```

\gls@alphpage

```

4634 \newcommand*{\gls@alphpage}{\@alph\c@page}

```

```

\gls@Alphpage
4635 \newcommand*{\gls@Alphpage}{\@Alph\c@page}

\gls@numberpage
4636 \newcommand*{\gls@numberpage}{\number\c@page}

\gls@romanpage
4637 \newcommand*{\gls@romanpage}{\romannumeral\c@page}

\gls@Romanpage
4638 \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.
4639 \newcommand*{\@do@wrglossary}[1]{%
4640   \begingroup
      First a bit of hackery to prevent premature expansion of \c@page. Store original
      definitions:
4641   \let\orgthe\the
4642   \let\orgnumber\number
4643   \let\orgromannumeral\romannumeral
4644   \let\orgalph\@alph
4645   \let\orgAlph\@Alph
4646   \let\orgRoman\@Roman
      Redefine:
4647   \def\the##1{%
4648     \ifx##1\c@page \gls@numberpage\else\orgthe##1\fi}%
4649   \def\number##1{%
4650     \ifx##1\c@page \gls@numberpage\else\orgnumber##1\fi}%
4651   \def\romannumeral##1{%
4652     \ifx##1\c@page \gls@romanpage\else\orgromannumeral##1\fi}%
4653   \def\@Roman##1{%
4654     \ifx##1\c@page \gls@Romanpage\else\orgRoman##1\fi}%
4655   \def\@alph##1{%
4656     \ifx##1\c@page \gls@alphpage\else\orgalph##1\fi}%
4657   \def\@Alph##1{%
4658     \ifx##1\c@page \gls@Alphpage\else\orgAlph##1\fi}%
      Prevent expansion:
4659   \gls@disablepagerefexpansion
      Now store location in \glslocref:
4660   \protected@xdef\glslocref{\the\glsentrycounter}%
4661   \endgroup
      Escape any special characters
4662   \gls@checkmkidxchars\glslocref

```


Check if the hyper-location is the same as the location and set the hyper prefix.

```

4663 \expandafter\ifx\theHglSentrycounter\theHglSentrycounter\relax
4664 \def\@glo@counterprefix{%
4665 \else
4666 \protected@edef\@glsHlocref{\theHglSentrycounter}%
4667 \@gls@checkmkidxchars\@glsHlocref
4668 \edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix
4669 {\@glslocref}{\@glsHlocref}%
4670 }%
4671 \@do@gls@getcounterprefix
4672 \fi

```

De-tok label if required

```

4673 \edef\@gls@label{\glsdetoklabel{#1}}%

```

Write the information to file:

```

4674 \@do@wrglossary
4675 }

```

\@do@wrglossary

```

4676 \newcommand*\@do@wrglossary{%

```

Determine whether to use xindy or makeindex syntax

```

4677 \ifglxindy

```

Need to determine if the formatting information starts with a (or) indicating a range.

```

4678 \expandafter\@glo@check@mkidxrangechar\@glsnumberformat\@nil
4679 \def\@glo@range{%
4680 \expandafter\if\@glo@prefix(\relax
4681 \def\@glo@range{:open-range}%
4682 \else
4683 \expandafter\if\@glo@prefix)\relax
4684 \def\@glo@range{:close-range}%
4685 \fi
4686 \fi

```

Write to the glossary file using xindy syntax.

```

4687 \gls@glossary{\csname glo@\@gls@label @type\endcsname}{%
4688 (indexentry :tkey (\csname glo@\@gls@label @index\endcsname)
4689 :locref \string{\@glo@counterprefix}{\@glslocref}\string" %
4690 :attr \string\@gls@counter\@glo@suffix\string"
4691 \@glo@range
4692 )
4693 }%
4694 \else

```

Convert the format information into the format required for makeindex

```

4695 \@set@glo@numformat{\@glo@numfmt}{\@gls@counter}{\@glsnumberformat}%
4696 {\@glo@counterprefix}%

```

Write to the glossary file using makeindex syntax.

```

4697 \gls@glossary{\csname glo@\gls@label @type\endcsname}{%
4698 \string\glossaryentry{\csname glo@\gls@label @index\endcsname
4699 \@gls@encapchar\@glo@numfmt}{\@gls@locref}}%
4700 \fi
4701 }

```

`\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.)

```

4702 \newcommand*\@gls@getcounterprefix[2]{%
4703 \edef\@gls@thisloc{#1}\edef\@gls@thisHloc{#2}%
4704 \ifx\@gls@thisloc\@gls@thisHloc
4705 \def\@glo@counterprefix{%
4706 \else
4707 \def\@gls@get@counterprefix##1.#1##2\end@getprefix{%
4708 \def\@glo@tmp{##2}%
4709 \ifx\@glo@tmp\@empty
4710 \def\@glo@counterprefix{%
4711 \else
4712 \def\@glo@counterprefix{##1}%
4713 \fi
4714 }%
4715 \@gls@get@counterprefix#2.#1\end@getprefix

```

Warn if no prefix can be formed.

```

4716 \ifx\@glo@counterprefix\@empty
4717 \GlossariesWarning{Hyper target ‘#2’ can’t be formed by
4718 prefixing location ‘#1’. You need to modify the
4719 definition of \string\theH\@gls@counter otherwise you
4720 will get the warning: “name{\@gls@counter.#1}’ has been
4721 referenced but does not exist”}%
4722 \fi
4723 \fi
4724 }

```

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 `[\<tag>]{\<list>}`, where `\<tag>` is a tag such as “see” and `\<list>` is a list of labels.

```

4725 \newcommand{\@do@seeglossary}[2]{%
4726 \def\@gls@xref{#2}%
4727 \@onelevel@sanitize\@gls@xref
4728 \@gls@checkmkidxchars\@gls@xref
4729 \ifglxsindy

```

```

4730 \gls@glossary{\csname glo@#1@type\endcsname}{%
4731   (indexentry
4732     :tkey (\csname glo@#1@index\endcsname)
4733     :xref (\string"\@gls@xref\string")
4734     :attr \string"see\string"
4735   )
4736 }%
4737 \else
4738 \gls@glossary{\csname glo@#1@type\endcsname}{%
4739 \string\glossaryentry{\csname glo@#1@index\endcsname
4740 \@gls@encapchar glsseeformat\@gls@xref}{Z}}%
4741 \fi
4742 }

```

`\@gls@fixbraces` If no optional argument is specified, list needs to be enclosed in a set of braces.

```

4743 \def\@gls@fixbraces#1#2#3\@nil{%
4744   \ifx#2[\relax
4745     \@gls@fixbraces#1#2#3\@end@fixbraces
4746   \else
4747     \def#1{{#2#3}}%
4748   \fi
4749 }

```

`\@@gls@fixbraces`

```

4750 \def\@@gls@fixbraces#1[#2]#3\@end@fixbraces{%
4751   \def#1{[#2]{#3}}%
4752 }

```

`\glssee` `\glssee{<label>}{<cross-ref list>}`

```

4753 \DeclareRobustCommand*\glssee[3][\seename]{%
4754   \@do@seeglossary{#2}{[#1]{#3}}}
4755 \newcommand*\@glssee[3][\seename]{%
4756   \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.

```

4757 \DeclareRobustCommand*\glsseeformat[3][\seename]{%
4758   \emph{#1} \glsseelist{#2}}

```

`\glsseelist` `\glsseelist{<list>}` formats list of entry labels.

```

4759 \DeclareRobustCommand*\glsseelist[1]{%

```

If there is only one item in the list, set the last separator to do nothing.

```

4760 \let\@gls@dolast\relax

```

Don't display separator on the first iteration of the loop

```

4761 \let\@gls@donext\relax

```

Iterate through the labels

```

4762 \@for\@gls@thislabel:=#1\do{%

```

Check if on last iteration of loop

```
4763 \ifx\@xfor@nextelement\@nnil
4764 \@gls@dolast
4765 \else
4766 \@gls@donext
4767 \fi
```

Display the entry for this label. (Expanding label as it's a temporary control sequence that's used elsewhere.)

```
4768 \expandafter\glsseeitem\expandafter{\@gls@thislabel}%
```

Update separators

```
4769 \let\@gls@dolast\glsseelastsep
4770 \let\@gls@donext\glsseesep
4771 }%
4772 }
```

`\glsseelastsep` Separator to use between penultimate and ultimate entries in a cross-referencing list.

```
4773 \newcommand*{\glsseelastsep}{\space\andname\space}
```

`\glsseesep` Separator to use between entires in a cross-referencing list.

```
4774 \newcommand*{\glsseesep}{, }
```

`\glsseeitem` `\glsseeitem{<label>}` formats individual entry in a cross-referencing list.

```
4775 \DeclareRobustCommand*{\glsseeitem}[1]{\glshyperlink[\glsseeitemformat{#1}]{#1}}
```

`\glsseeitemformat` As from v3.0, default is to use `\glentrytext` instead of `\glentryname`. (To avoid problems with the name key being sanitized.)

```
4776 \newcommand*{\glsseeitemformat}[1]{\glentrytext{#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.

```
4777 \newcommand*{\gls@save@numberlist}[1]{%
4778 \ifglssavenumberlist
4779 \toks@{#1}%
4780 \edef\@do@writeaux@info{%
4781 \noexpand\csgdef{glo@\glscurrententrylabel @numberlist}{\the\toks@}%
4782 }%
4783 \@onelevel@sanitize\@do@writeaux@info
4784 \protected@write\@auxout{}\@do@writeaux@info}%
```

```

4785 \fi
4786 }

\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.)
4787 \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.
4788 \ifcsundef{printglossary}{}%
4789 {%
    If \printglossary is already defined, issue a warning and undefine it.
4790 \@gls@warnonglossdefined
4791 \undef\printglossary
4792 }

    \printglossary has an optional argument. The default value is to set the glos-
    sary type to the main glossary.
4793 \newcommand*{\printglossary}[1][type=\glsdefaulttype]{%
4794 \@printglossary{#1}{\@print@glossary}%
4795 }

    The \printglossaries command will do \printglossary for each glos-
    sary 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 con-
    tents 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
4796 \newcommand*{\printglossaries}{%
4797 \forallglossaries{\@glo@type}{\printglossary[type=\@glo@type]}}%
4798 }

\printnoidxglossary Provide an alternative to \printglossary that doesn't require an external in-
                    dexing application. Entries won't be sorted and the location list will be empty.
4799 \newcommand*{\printnoidxglossary}[1][type=\glsdefaulttype]{%
4800 \@printglossary{#1}{\@print@noidx@glossary}%
4801 }

\printnoidxglossaries Analogous to \printglossaries
4802 \newcommand*{\printnoidxglossaries}{%
4803 \forallglossaries{\@glo@type}{\printnoidxglossary[type=\@glo@type]}}%
4804 }

```

`@printgloss@setsort` Initialise to do nothing.

```
4805 \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.

```
4806 \newcommand{\@printglossary}[2]{%
  Set up defaults.
  4807   \def\@glo@type{\glsdefaulttype}%
  4808   \def\glossarytitle{\csname @glo@type \@glo@type @title\endcsname}%

  4809   \def\glossarytoctitle{\glossarytitle}%
  4810   \let\org@glossarytitle\glossarytitle
  4811   \def\@glossarystyle{}%
  4812   \def\gls@dotoc@title{\glssettoctitle{\@glo@type}}%

  Store current value of \glossaryentrynumbers. (This may be changed via the
  optional argument)
  4813   \let\@org@glossaryentrynumbers\glossaryentrynumbers

  Localise the effects of the optional argument
  4814   \bgroup

  Activate or deactivate sort key:
  4815   \@printgloss@setsort

  Determine settings specified in the optional argument.
  4816   \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)
  4817   \ifx\glossarytitle\org@glossarytitle
  4818   \else
  4819     \expandafter\let\csname @glo@type \@glo@type @title\endcsname
  4820       \glossarytitle
  4821   \fi

  Allow a high-level user command to indicate the current glossary
  4822   \let\currentglossary\@glo@type

  Enable individual number lists to be suppressed.
  4823   \let\org@glossaryentrynumbers\glossaryentrynumbers
  4824   \let\glsnonextpages\@glsnonextpages

  Enable individual number list to be activated:
  4825   \let\glsnextpages\@glsnextpages

  Enable suppression of description terminators.
  4826   \let\nopostdesc\@nopostdesc

  Set up the entry for the TOC
  4827   \gls@dotoc@title
```

Set the glossary style

```
4828 \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):

```
4829 \let\gls@org@glossaryentryfield\glossentry
4830 \let\gls@org@glossarysubentryfield\subglossentry
4831 \renewcommand{\glossentry}[1]{%
4832   \xdef\glscurrententrylabel{\glsdetoklabel{##1}}%
4833   \gls@org@glossaryentryfield{##1}%
4834 }%
4835 \renewcommand{\subglossentry}[2]{%
4836   \xdef\glscurrententrylabel{\glsdetoklabel{##2}}%
4837   \gls@org@glossarysubentryfield{##1}{##2}%
4838 }%
```

Now do the handler macro that deals with the actual glossary:

```
4839 #2%
```

End the current scope

```
4840 \egroup
```

Reset `\glossaryentrynumbers`

```
4841 \global\let\glossaryentrynumbers\@org@glossaryentrynumbers
```

Suppress warning about no `\printglossary`

```
4842 \global\let\warn@noprntglossary\relax
4843 }
```

`\@print@glossary` Internal workings of `\printglossary` dealing with reading the external file.

```
4844 \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.)

```
4845 \makeatletter
```

Input the glossary file, if it exists.

```
4846 \@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.

```
4847 \IfFileExists{\jobname.\csname @glotype@\@glo@type @in\endcsname}%
4848 {}%
4849 {\null}%
```

If `xindy` is being used, need to write the language dependent information to the `.aux` file for `makeglossaries`.

```
4850 \ifglxindy
```

```

4851 \ifcsundef{@xdy@\@glo@type @language}%
4852 {%
4853     \edef\@do@auxoutstuff{%
4854         \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.

```

4855         \noexpand\immediate\noexpand\write\@auxout{%
4856             \string\providecommand\string\@xdylanguage[2]{}%
4857         \noexpand\immediate\noexpand\write\@auxout{%
4858             \string\@xdylanguage{\@glo@type}{\@xdy@main@language}}%
4859     }%
4860 }%
4861 }%
4862 {%
4863     \edef\@do@auxoutstuff{%
4864         \noexpand\AtEndDocument{%
4865             \noexpand\immediate\noexpand\write\@auxout{%
4866                 \string\providecommand\string\@xdylanguage[2]{}%
4867             \noexpand\immediate\noexpand\write\@auxout{%
4868                 \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type
4869                     @language\endcsname}}%
4870         }%
4871     }%
4872 }%
4873 \do@auxoutstuff
4874 \edef\@do@auxoutstuff{%
4875     \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.

```

4876         \noexpand\immediate\noexpand\write\@auxout{%
4877             \string\providecommand\string\@gls@codepage[2]{}%
4878         \noexpand\immediate\noexpand\write\@auxout{%
4879             \string\@gls@codepage{\@glo@type}{\gls@codepage}}%
4880     }%
4881 }%
4882 \do@auxoutstuff
4883 \fi

```

Activate warning if \makeglossaries hasn't been used.

```

4884 \renewcommand*{\@warn@nomakeglossaries}{%
4885     \GlossariesWarningNoLine{\string\makeglossaries\space
4886     hasn't been used,^^Jthe glossaries will not be updated}%
4887 }%
4888 }

```

The sort macros all have the syntax:

`\@glo@sortmacro@<order>{<type>}`

where *<order>* is the sort order as specified by the sort key and *<type>* is the glossary type. (The referenced entry list is stored in `\@glsref@<type>`). The actual sorting is done by `\@glo@sortentries{<handler>}{<type>}`.

`\@glo@sortentries`

```

4889 \newcommand*{\@glo@sortentries}[2]{%
4890   \def\@glo@sortinglist{}%
4891   \def\@glo@sortinghandler{#1}%
4892   \edef\@glo@type{#2}%
4893   \forlistcsloop{\@glo@do@sortentries}{@glsref@#2}%
4894   \csdef{@glsref@#2}{}%
4895   \@for\@this@label:=\@glo@sortinglist\do{%
      Has this entry already been added?
4896     \xifinlistcs{\@this@label}{@glsref@#2}%
4897     {}%
4898     {%
4899       \listcsxadd{@glsref@#2}{\@this@label}%
4900     }%
4901     \ifcsdef{@glo@sortingchildren@ \@this@label}%
4902     {%
4903       \@glo@addchildren{#2}{\@this@label}%
4904     }%
4905     {}%
4906   }%
4907 }
```

`\@glo@addchildren`

`\@glo@addchildren{<type>}{<parent>}`

```

4908 \newcommand*{\@glo@addchildren}[2]{%
      Scope to allow nesting.
4909   \bgroup
4910     \letcs{\@glo@childlist}{@glo@sortingchildren@#2}%
4911     \@for\@this@childlabel:=\@glo@childlist\do
4912     {%
      Check this label hasn't already been added.
4913       \xifinlistcs{\@this@childlabel}{@glsref@#1}%
4914       {}%
4915       {%
4916         \listcsxadd{@glsref@#1}{\@this@childlabel}%
4917       }%
      Does this child have children?
4918       \ifcsdef{@glo@sortingchildren@ \@this@childlabel}%
4919       {%
```

```

4920      \@glo@addchildren{#1}{\@this@childlabel}%
4921      }%
4922      {%
4923      }%
4924      }%
4925      \egroup
4926 }

```

@glo@do@sortentries

```

4927 \newcommand*{\@glo@do@sortentries}[1]{%
4928   \ifglshasparent{#1}%
4929   {%
      This entry has a parent, so add it to the child list
4930   \edef\@glo@parent{\csuse{glo@\glsetoklabel{#1}@parent}}%
4931   \ifcsundef{glo@sortingchildren@\@glo@parent}%
4932   {%
4933     \csdef{glo@sortingchildren@\@glo@parent}{}%
4934   }%
4935   {%
4936     \expandafter\@glo@sortedinsert
4937     \csname glo@sortingchildren@\@glo@parent\endcsname{#1}%

```

Has the parent been added?

```

4938     \xifinlistcs{\@glo@parent}{\glstrref{\@glo@type}}%
4939     {%

```

Yes, it has so do nothing.

```

4940     }%
4941     {%

```

No, it hasn't so add it now.

```

4942     \expandafter\@glo@do@sortentries\expandafter{\@glo@parent}%
4943     }%
4944     }%
4945     {%
4946     \@glo@sortedinsert{\@glo@sortinglist}{#1}%
4947     }%
4948 }

```

\@glo@sortedinsert `\@glo@sortedinsert{<list>}{<entry label>}`

Insert into list.

```

4949 \newcommand*{\@glo@sortedinsert}[2]{%
4950   \dtl@insertinto{#2}{#1}{\@glo@sortinghandler}%
4951 }%

```

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

```
4952 \newcommand*{\@glo@sorthandler@word}[2]{%
4953   \letcs\@gls@sort@A{glo\glsdetoklabel{#1}@sort}%
4954   \letcs\@gls@sort@B{glo\glsdetoklabel{#2}@sort}%
4955   \edef\glo@do@compare{%
4956     \noexpand\dtlwordindexcompare{\noexpand\dtl@sortresult}%
4957     {\expandonce\@gls@sort@B}%
4958     {\expandonce\@gls@sort@A}%
4959   }%
4960   \glo@do@compare
4961 }
```

@sorthandler@letter

```
4962 \newcommand*{\@glo@sorthandler@letter}[2]{%
4963   \letcs\@gls@sort@A{glo\glsdetoklabel{#1}@sort}%
4964   \letcs\@gls@sort@B{glo\glsdetoklabel{#2}@sort}%
4965   \edef\glo@do@compare{%
4966     \noexpand\dtlletterindexcompare{\noexpand\dtl@sortresult}%
4967     {\expandonce\@gls@sort@B}%
4968     {\expandonce\@gls@sort@A}%
4969   }%
4970   \glo@do@compare
4971 }
```

lo@sorthandler@case Case-sensitive sort.

```
4972 \newcommand*{\@glo@sorthandler@case}[2]{%
4973   \letcs\@gls@sort@A{glo\glsdetoklabel{#1}@sort}%
4974   \letcs\@gls@sort@B{glo\glsdetoklabel{#2}@sort}%
4975   \edef\glo@do@compare{%
4976     \noexpand\dtlcompare{\noexpand\dtl@sortresult}%
4977     {\expandonce\@gls@sort@B}%
4978     {\expandonce\@gls@sort@A}%
4979   }%
4980   \glo@do@compare
4981 }
```

@sorthandler@nocase Case-insensitive sort.

```
4982 \newcommand*{\@glo@sorthandler@nocase}[2]{%
4983   \letcs\@gls@sort@A{glo\glsdetoklabel{#1}@sort}%
4984   \letcs\@gls@sort@B{glo\glsdetoklabel{#2}@sort}%
4985   \edef\glo@do@compare{%
4986     \noexpand\dtlicompare{\noexpand\dtl@sortresult}%
4987     {\expandonce\@gls@sort@B}%
4988     {\expandonce\@gls@sort@A}%
4989   }%
4990   \glo@do@compare
4991 }
```

@glo@sortmacro@word Sort macro for ‘word’

```
4992 \newcommand*{\@glo@sortmacro@word}[1]{%
4993   \ifdefstring{\@glo@default@sorttype}{standard}%
4994   {%
4995     \@glo@sortentries{\@glo@sorthandler@word}{#1}%
4996   }%
4997   {%
4998     \PackageError{glossaries}{Conflicting sort options:^^J
4999     \string\usepackage[sort=\@glo@default@sorttype]{glossaries}^^J
5000     \string\printnoidxglossary[sort=word]}{}%
5001   }%
5002 }
```

lo@sortmacro@letter Sort macro for ‘letter’

```
5003 \newcommand*{\@glo@sortmacro@letter}[1]{%
5004   \ifdefstring{\@glo@default@sorttype}{standard}%
5005   {%
5006     \@glo@sortentries{\@glo@sorthandler@letter}{#1}%
5007   }%
5008   {%
5009     \PackageError{glossaries}{Conflicting sort options:^^J
5010     \string\usepackage[sort=\@glo@default@sorttype]{glossaries}^^J
5011     \string\printnoidxglossary[sort=letter]}{}%
5012   }%
5013 }
```

@sortmacro@standard Sort macro for ‘standard’. (Use either ‘word’ or ‘letter’ order.)

```
5014 \newcommand*{\@glo@sortmacro@standard}[1]{%
5015   \ifdefstring{\@glo@default@sorttype}{standard}%
5016   {%
5017     \ifcsdef{\@glo@sorthandler@\glsorder}%
5018     {%
5019       \@glo@sortentries{\csuse{\@glo@sorthandler@\glsorder}}{#1}%
5020     }%
5021     {%
5022       \PackageError{glossaries}{Unknown sort handler ‘\glsorder’}{}%
5023     }%
5024   }%
5025   {%
5026     \PackageError{glossaries}{Conflicting sort options:^^J
5027     \string\usepackage[sort=\@glo@default@sorttype]{glossaries}^^J
5028     \string\printnoidxglossary[sort=standard]}{}%
5029   }%
5030 }
```

@glo@sortmacro@case Sort macro for ‘case’

```
5031 \newcommand*{\@glo@sortmacro@case}[1]{%
5032   \ifdefstring{\@glo@default@sorttype}{standard}%
5033   {%
```

```

5034 \glo@sortentries{\glo@sorthandler@case}{#1}%
5035 }%
5036 {%
5037 \PackageError{glossaries}{Conflicting sort options:^^J
5038 \string\usepackage[sort=\glo@default@sorttype]{glossaries}^^J
5039 \string\printnoidxglossary[sort=case]}{}%
5040 }%
5041 }

\lo@sortmacro@nocase Sort macro for 'nocase'

5042 \newcommand*\glo@sortmacro@nocase[1]{%
5043 \ifdefstring{\glo@default@sorttype}{standard}%
5044 {%
5045 \glo@sortentries{\glo@sorthandler@nocase}{#1}%
5046 }%
5047 {%
5048 \PackageError{glossaries}{Conflicting sort options:^^J
5049 \string\usepackage[sort=\glo@default@sorttype]{glossaries}^^J
5050 \string\printnoidxglossary[sort=nocase]}{}%
5051 }%
5052 }

\glo@sortmacro@def Sort macro for 'def'. The order of definition is given in \glo@list@<type>.

5053 \newcommand*\glo@sortmacro@def[1]{%
5054 \def\glo@sortinglist{}%
5055 \forlslentries[#1]{\gls@thislabel}%
5056 {%
5057 \xifinlistcs{\gls@thislabel}{\glsref@#1}%
5058 {%
5059 \listcsadd{\glo@sortinglist}{\gls@thislabel}%
5060 }%
5061 {%
5062 }%
5063 }%
5064 \cslet{\glsref@#1}{\glo@sortinglist}%
5065 }

\lo@sortmacro@def@do This won't include parent entries that haven't been referenced.

5066 \newcommand*\glo@sortmacro@def@do[1]{%
5067 \ifinlistcs{#1}{\glsref@\glo@type}%
5068 {}%
5069 {%
5070 \listcsadd{\glsref@\glo@type}{#1}%
5071 }%
5072 \ifcsdef{\glo@sortingchildren@#1}%
5073 {%
5074 \glo@addchildren{\glo@type}{#1}%

```

```

5075 }%
5076 {}%
5077 }

\@glo@sortmacro@use Sort macro for 'use'. (No sorting is required, as the entries are already in order
of use, so do nothing.)
5078 \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.
5079 \newcommand*{\@print@noidx@glossary}{%
5080 \ifcsdef{@glsref@ \@glo@type}%
5081 {%
Sort the entries:
5082 \ifcsdef{@glo@sortmacro@ \@glo@sorttype}%
5083 {%
5084 \csuse{@glo@sortmacro@ \@glo@sorttype}{\@glo@type}%
5085 }%
5086 {%
5087 \PackageError{glossaries}{Unknown sort handler '@@glo@sorttype'}{ }%
5088 }%

Do the glossary heading and preamble
5089 \glossarysection[\glossarytoctitle]{\glossarytitle}%
5090 \glossarypreamble
5091 \begin{theglossary}%
5092 \glossaryheader
5093 \glsresetentrylist
5094 \def\@gls@currentlettergroup{ }%

Iterate through the entries.
5095 \forlistcsloop{\@gls@noidx@do}{@glsref@ \@glo@type}%

Finally end the glossary and do the postamble:
5096 \end{theglossary}%
5097 \glossarypostamble
5098 }%
5099 {%
5100 \@gls@noref@warn{\@glo@type}%
5101 }%
5102 }

\glo@grabfirst
5103 \def\glo@grabfirst#1#2\@nil{%
5104 \def\@gls@firsttok{#1}%

```

```

5105 \ifdefempty\@gls@firsttok
5106 {%
5107   \def\@glo@thislettergrp{0}%
5108 }%
5109 {%

  Sanitize it:
5110   \@onelevel@sanitize\@gls@firsttok

  Fetch the first letter:
5111   \expandafter\@glo@grabfirst\@gls@firsttok{}\}\@nil
5112 }%
5113 }

```

\@glo@grabfirst

```

5114 \def\@glo@grabfirst#1#2\@nil{%
5115   \ifdefempty\@glo@thislettergrp
5116   {%
5117     \def\@glo@thislettergrp{glssymbols}%
5118   }%
5119   {%
5120     \count@=\uccode'#1\relax
5121     \ifnum\count@=0\relax
5122       \def\@glo@thislettergrp{glssymbols}%
5123     \else
5124       \ifdefstring\@glo@sorttype{case}%
5125       {%
5126         \count@='#1\relax
5127       }%
5128       {%
5129       }%
5130       \edef\@glo@thislettergrp{\the\count@}%
5131     \fi
5132   }%
5133 }

```

\@gls@noidx@do Handler for list iteration used by \@print@noidx@glossary. The argument is the entry label. This only allows one sublevel.

```

5134 \newcommand{\@gls@noidx@do}[1]{%

  Get this entry's location list
5135   \global\letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%

  Does this entry have a parent?
5136   \ifglshasparent{#1}%
5137   {%

    Has a parent.
5138     \gls@level=\csuse{glo@\glsdetoklabel{#1}@level}\relax
5139     \ifdefvoid{\@gls@loclist}
5140     {%

```

```

5141     \subglossentry{\gls@level}{#1}{}%
5142   }%
5143   {%
5144     \subglossentry{\gls@level}{#1}%
5145     {%
5146       \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
5147     }%
5148   }%
5149 }%
5150 {%

```

Doesn't have a parent Get this entry's sort key

```

5151   \letcs{\@gls@sort}{glo@glsdetoklabel{#1}@sort}%

```

Fetch the first letter:

```

5152   \expandafter\glo@grabfirst\@gls@sort{}{}\@nil
5153   \ifdefequal{\@glo@thislettergrp}{\@gls@currentlettergroup}%
5154   {}%
5155   {%

```

Do the group header:

```

5156   \ifdefempty{\@gls@currentlettergroup}{\@gls@groupskip}%
5157   \gls@groupheading{\@glo@thislettergrp}%
5158   }%
5159   \let\@gls@currentlettergroup\@glo@thislettergrp

```

Do this entry:

```

5160   \ifdefvoid{\@gls@loclist}
5161   {%
5162     \glossentry{#1}{}%
5163   }%
5164   {%
5165     \glossentry{#1}%
5166     {%
5167       \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
5168     }%
5169   }%
5170 }%
5171 }

```

\glsnoidxloclist `\glsnoidxloclist{<list cs>}`

Display location list.

```

5172 \newcommand*{\glsnoidxloclist}[1]{%
5173   \def\@gls@noidxloclist@sep{}%
5174   \def\@gls@noidxloclist@prev{}%
5175   \forlistloop{\glsnoidxloclisthandler}{#1}%
5176 }

```


noidxloclisthandler Handler for location list iterator.

```
5177 \newcommand*{\glsnoidxloclisthandler}[1]{%
5178   \ifdefstring{\@gls@noidxloclist@prev}{#1}%
5179   {%
      Same as previous location so skip.
5180   }%
5181   {%
5182     \@gls@noidxloclist@sep
5183     #1%
5184     \def\@gls@noidxloclist@sep{\delimN}%
5185     \def\@gls@noidxloclist@prev{#1}%
5186   }%
5187 }
```

splayloclisthandler Handler for location list iterator when used with \glsdisplaynumberlist.

```
5188 \newcommand*{\glsnoidxdisplayloclisthandler}[1]{%
5189   \ifdefstring{\@gls@noidxloclist@prev}{#1}%
5190   {%
      Same as previous location so skip.
5191   }%
5192   {%
5193     \@gls@noidxloclist@sep
5194     \@gls@noidxloclist@prev
5195     \def\@gls@noidxloclist@prev{#1}%
5196   }%
5197 }
```

\glsnoidxdisplayloc `\glsnoidxdisplayloc{<prefix>}{<counter>}{<format>}{<location>}`

Display a location in the location list.

```
5198 \newcommand*\glsnoidxdisplayloc[4]{%
5199   \setentrycounter[#1]{#2}%
5200   \csuse{#3}{#4}%
5201 }
```

\@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.

```
5202 \newcommand*\@gls@reference[3]{%
      Add to label list
5203   \glsdoifexistsorwarn{#2}%
5204   {%
5205     \ifcsundef{\@glsref@#1}{\csgdef{\@glsref@#1}{}}{}%

```

```

5206 \ifinlistcs{#2}{@glsref@#1}%
5207 {}%
5208 {\listcsgadd{@glsref@#1}{#2}}%
    Add to location list
5209 \ifcsundef{glo@glstdetoklabel{#2}@loclist}%
5210 {\csgdef{glo@glstdetoklabel{#2}@loclist}{}}%
5211 {}%
5212 \listcsgadd{glo@glstdetoklabel{#2}@loclist}{#3}%
5213 }%
5214 }

```

The keys that can be used in the optional argument to `\printglossary` or `\printnoidxglossary` are as follows: The `type` key sets the glossary type.

```

5215 \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`.

```

5216 \define@key{printgloss}{title}{%
5217 \def@glossarytitle{#1}%
5218 \let@gls@dotoc@title\relax
5219 }

```

The `toctitle` sets the text used for the relevant entry in the table of contents.

```

5220 \define@key{printgloss}{toctitle}{%
5221 \def@glossarytoctitle{#1}%
5222 \let@gls@dotoc@title\relax
5223 }

```

The `style` key sets the glossary style (but only for the given glossary).

```

5224 \define@key{printgloss}{style}{%
5225 \ifcsundef{@glsstyle@#1}%
5226 {%
5227 \PackageError{glossaries}%
5228 {Glossary style ‘#1’ undefined}{}%
5229 }%
5230 {%
5231 \def@glossarystyle{\setglossentrycompatibility
5232 \csname @glsstyle@#1\endcsname}%
5233 }%
5234 }

```

The `numberedsection` key determines if this glossary should be in a numbered section.

```

5235 \define@choicekey{printgloss}{numberedsection}[\val\nr]{%
5236 false,nolabel,autolabel,nameref}[nolabel]{%
5237 \ifcase\nr\relax
5238 \renewcommand*{\@@glossarysecstar}{*}%
5239 \renewcommand*{\@@glossaryseclabel}{}%
5240 \or
5241 \renewcommand*{\@@glossarysecstar}{}%

```

```

5242 \renewcommand*{\@@glossaryseclabel}{}%
5243 \or
5244 \renewcommand*{\@@glossarysecstar}{}%
5245 \renewcommand*{\@@glossaryseclabel}{\label{\glsautoprefix\@glo@type}}%
5246 \or
5247 \renewcommand*{\@@glossarysecstar}{*}%
5248 \renewcommand*{\@@glossaryseclabel}{%
5249 \protected@edef\@currentlabelname{\glossarytoctitle}%
5250 \label{\glsautoprefix\@glo@type}}%
5251 \fi
5252 }

```

The `nogroupskip` key determines whether or not there should be a vertical gap between glossary groups.

```

5253 \define@choicekey{printgloss}{nogroupskip}{true,false}[true]{%
5254 \csuse{glsnogroupskip#1}%
5255 }

```

The `nopostdot` key has the same effect as the package option of the same name.

```

5256 \define@choicekey{printgloss}{nopostdot}{true,false}[true]{%
5257 \csuse{glsnopostdot#1}%
5258 }

```

The `entrycounter` key is the same as the package option but localised to the current glossary.

```

5259 \define@choicekey{printgloss}{entrycounter}{true,false}[true]{%
5260 \csuse{glsentrycounter#1}%
5261 \ifglsentrycounter
5262 \ifx\@gls@counterwithin\@empty
5263 \newcounter{glossaryentry}%
5264 \else
5265 \newcounter{glossaryentry}[\@gls@counterwithin]%
5266 \fi
5267 \def\theHglossaryentry{\currentglossary.\theglossaryentry}%
5268 \renewcommand*{\glsresetentrycounter}{%
5269 \setcounter{glossaryentry}{0}%
5270 }%
5271 \renewcommand*{\glsstepentry}[1]{%
5272 \refstepcounter{glossaryentry}%
5273 \label{glsentry-\glsdetoklabel{##1}}%
5274 }%
5275 \renewcommand*{\glsentrycounterlabel}{\theglossaryentry.\space}%
5276 \renewcommand*{\glsentryitem}[1]{%
5277 \glsstepentry{##1}\glsentrycounterlabel
5278 }%
5279 \else
5280 \renewcommand*{\glsresetentrycounter}{}%
5281 \renewcommand*{\glsstepentry}[1]{%
5282 \renewcommand*{\glsentrycounterlabel}{}%

```

```

5283 \renewcommand*{\glsubentryitem}[1]{\glsresetsubentrycounter}
5284 \fi
5285 }

```

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.

```

5286 \define@choicekey{printgloss}{subentrycounter}{true,false}[true]{%
5287 \csuse{glssubentrycounter#1}%
5288 \ifglssubentrycounter
5289 \ifundef\c@glossarysubentry
5290 {%
5291 \ifglsubentrycounter
5292 \newcounter{glossarysubentry}[glossaryentry]%
5293 \else
5294 \newcounter{glossarysubentry}
5295 \fi
5296 }{}%
5297 \renewcommand*{\glsstepsubentry}[1]{%
5298 \edef\currentglssubentry{\glsdetoklabel{##1}}%
5299 \refstepcounter{glossarysubentry}%
5300 \label{glsubentry-\currentglssubentry}%
5301 }%
5302 \renewcommand*{\glsresetsubentrycounter}{%
5303 \setcounter{glossarysubentry}{0}%
5304 }%
5305 \renewcommand*{\glssubentryitem}[1]{%
5306 \glsstepsubentry{##1}\glssubentrycounterlabel
5307 }%
5308 \renewcommand*{\glssubentrycounterlabel}{\theglossarysubentry}\space}%
5309 \def\theHglossarysubentry{\currentglssubentry.\theglossarysubentry}
5310 \else
5311 \renewcommand*{\glssubentryitem}[1]{}%
5312 \renewcommand*{\glsstepsubentry}[1]{}%
5313 \renewcommand*{\glsresetsubentrycounter}{}%
5314 \renewcommand*{\glssubentrycounterlabel}{}%
5315 \fi
5316 }

```

The nonnumberlist key determines if this glossary should have a number list.

```

5317 \define@boolkey{printgloss}[gls]{nonnumberlist}[true]{%
5318 \ifglsnonnumberlist
5319 \def\glossaryentrynumbers##1{}%
5320 \else
5321 \def\glossaryentrynumbers##1{##1}%
5322 \fi}

```

The sort key sets the glossary sort handler (\printnoidxglossary only).

```

5323 \define@key{printgloss}{sort}{\@glo@assign@sortkey{#1}}

```

`@no@assign@sortkey` Issue error if used with `\printglossary`

```

5324 \newcommand*{\@glo@no@assign@sortkey}[1]{%
5325   \PackageError{glossaries}{‘sort’ key not permitted with
5326   \string\printglossary}%
5327   {The ‘sort’ key may only be used with \string\printnoidxglossary}%
5328 }
```

`@glo@assign@sortkey` For use with `\printnoidxglossary`

```

5329 \newcommand*{\@glo@assign@sortkey}[1]{%
5330   \def\@glo@sorttype{#1}%
5331 }
```

`\@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.

```

5332 \newcommand*{\@glsnonextpages}{%
5333   \gdef\glossaryentrynumbers##1{%
5334     \glsresetentrylist
5335   }%
5336 }
```

`\@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.

```

5337 \newcommand*{\@glsnextpages}{%
5338   \gdef\glossaryentrynumbers##1{%
5339     ##1\glsresetentrylist}}

```

`\glsresetentrylist` Resets `\glossaryentrynumbers`

```

5340 \newcommand*{\glsresetentrylist}{%
5341   \global\let\glossaryentrynumbers\org@glossaryentrynumbers}

```

`\glsnonextpages` Outside of `\printglossary` this does nothing.

```

5342 \newcommand*{\glsnonextpages}{}

```

`\glsnextpages` Outside of `\printglossary` this does nothing.

```

5343 \newcommand*{\glsnextpages}{}

```

`glossaryentry` If the `entrycounter` package option has been used, define a counter to number each level 0 entry.

```

5344 \ifglentrycounter
5345   \ifx\@gls@counterwithin\@empty
5346     \newcounter{glossaryentry}

```

```

5347 \else
5348   \newcounter{glossaryentry}[\@gls@counterwithin]
5349 \fi
5350 \def\theHglossaryentry{\currentglossary.\theglossaryentry}
5351 \fi

```

glossarysubentry If the subentrycounter package option has been used, define a counter to number each level 1 entry.

```

5352 \ifglssubentrycounter
5353   \ifglentrycounter
5354     \newcounter{glossarysubentry}[glossaryentry]
5355   \else
5356     \newcounter{glossarysubentry}
5357   \fi
5358   \def\theHglossarysubentry{\currentglssubentry.\theglossarysubentry}
5359 \fi

```

resetsubentrycounter Resets the glossarysubentry counter.

```

5360 \ifglssubentrycounter
5361   \newcommand*{\glsresetsubentrycounter}{%
5362     \setcounter{glossarysubentry}{0}%
5363   }
5364 \else
5365   \newcommand*{\glsresetsubentrycounter}{}
5366 \fi

```

resetentrycounter Resets the glossaryentry counter.

```

5367 \ifglentrycounter
5368   \newcommand*{\glsresetentrycounter}{%
5369     \setcounter{glossaryentry}{0}%
5370   }
5371 \else
5372   \newcommand*{\glsresetentrycounter}{}
5373 \fi

```

\glsstepentry Advance the glossaryentry counter if in use. The argument is the label associated with the entry.

```

5374 \ifglentrycounter
5375   \newcommand*{\glsstepentry}[1]{%
5376     \refstepcounter{glossaryentry}%
5377     \label{glentry-\glsdetoklabel{#1}}%
5378   }
5379 \else
5380   \newcommand*{\glsstepentry}[1]{}
5381 \fi

```

\glsstepsubentry Advance the glossarysubentry counter if in use. The argument is the label associated with the subentry.

```

5382 \ifglssubentrycounter
5383   \newcommand*{\glstepsubentry}[1]{%
5384     \edef\currentglssubentry{\glsetoklabel{#1}}%
5385     \refstepcounter{glossarysubentry}%
5386     \label{glentry-\currentglssubentry}%
5387   }
5388 \else
5389   \newcommand*{\glstepsubentry}[1]{%
5390 \fi

```

`\glhrefentry` Reference the entry or sub-entry counter if in use, otherwise just do `\gl`.

```

5391 \ifglentrycounter
5392   \newcommand*{\glhrefentry}[1]{\ref{glentry-\glsetoklabel{#1}}}
5393 \else
5394   \ifglssubentrycounter
5395     \newcommand*{\glhrefentry}[1]{\ref{glentry-\glsetoklabel{#1}}}
5396   \else
5397     \newcommand*{\glhrefentry}[1]{\gl{#1}}
5398   \fi
5399 \fi

```

`glentrycounterlabel` Defines how to display the glossaryentry counter.

```

5400 \ifglentrycounter
5401   \newcommand*{\glentrycounterlabel}{\theglossaryentry.\space}
5402 \else
5403   \newcommand*{\glentrycounterlabel}{}
5404 \fi

```

`glsubentrycounterlabel` Defines how to display the glossarysubentry counter.

```

5405 \ifglssubentrycounter
5406   \newcommand*{\glsubentrycounterlabel}{\theglossarysubentry)\space}
5407 \else
5408   \newcommand*{\glsubentrycounterlabel}{}
5409 \fi

```

`\glentryitem` Step and display glossaryentry counter, if appropriate.

```

5410 \ifglentrycounter
5411   \newcommand*{\glentryitem}[1]{%
5412     \glstepentry{#1}\glentrycounterlabel
5413   }
5414 \else
5415   \newcommand*{\glentryitem}[1]{\glresetsubentrycounter}
5416 \fi

```

`\glssubentryitem` Step and display glossarysubentry counter, if appropriate.

```

5417 \ifglssubentrycounter
5418   \newcommand*{\glssubentryitem}[1]{%
5419     \glstepsubentry{#1}\glssubentrycounterlabel
5420   }

```

```

5421 \else
5422   \newcommand*{\glssubentryitem}[1]{%
5423 \fi

```

theglossary If the theglossary environment has already been defined, a warning will be issued. This environment should be redefined by glossary styles.

```

5424 \ifcsundef{theglossary}%
5425 {%
5426   \newenvironment{theglossary}{}{}%
5427 }%
5428 {%
5429   \@gls@warnontheglossdefined
5430   \renewenvironment{theglossary}{}{}%
5431 }

```

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`

```

5432 \newcommand*{\glossaryheader}{}

```

`\glstarget` `\glstarget{<label>}{<name>}`

Provide user interface to `\@glstarget` to make it easier to modify the glossary style in the document.

```

5433 \newcommand*{\glstarget}[2]{\@glstarget{\glolinkprefix#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>}`

```

5434 \providecommand*{\compatibleglossentry}[2]{%
5435   \toks@{#2}%
5436   \protected@edef\@do@glossentry{\noexpand\glossaryentryfield{#1}%
5437     {\noexpand\glsnamefont
5438       {\expandafter\expandonce\csname glo@#1@name\endcsname}}}%
5439     {\expandafter\expandonce\csname glo@#1@desc\endcsname}}%
5440     {\expandafter\expandonce\csname glo@#1@symbol\endcsname}}%
5441     {\the\toks@}%
5442   }%

```



```

5443 \@do@glossentry
5444 }

```

\glossentryname

```

5445 \newcommand*{\glossentryname}[1]{%
5446   \glsdoifexistsorwarn{#1}%
5447   {%
5448     \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
5449     \expandafter\glsnamefont\expandafter{\glo@name}%
5450   }%
5451 }

```

\Glossentryname

```

5452 \newcommand*{\Glossentryname}[1]{%
5453   \glsdoifexistsorwarn{#1}%
5454   {%
5455     \glsnamefont{\Glsentryname{#1}}%
5456   }%
5457 }

```

\glossentrydesc

```

5458 \newcommand*{\glossentrydesc}[1]{%
5459   \glsdoifexistsorwarn{#1}%
5460   {%
5461     \glentrydesc{#1}%
5462   }%
5463 }

```

\Glossentrydesc

```

5464 \newcommand*{\Glossentrydesc}[1]{%
5465   \glsdoifexistsorwarn{#1}%
5466   {%
5467     \Glsentrydesc{#1}%
5468   }%
5469 }

```

\glossentrysymbol

```

5470 \newcommand*{\glossentrysymbol}[1]{%
5471   \glsdoifexistsorwarn{#1}%
5472   {%
5473     \glentrysymbol{#1}%
5474   }%
5475 }

```

\Glossentrysymbol

```

5476 \newcommand*{\Glossentrysymbol}[1]{%
5477   \glsdoifexistsorwarn{#1}%
5478   {%

```

```

5479 \Glsentrysymbol{#1}%
5480 }%
5481 }

```

compatiblesubglossentry `\subglossentry{<level>}{<label>}{<page-list>}`

```

5482 \providecommand*\compatiblesubglossentry}[3]{%
5483   \toks@{#3}%
5484   \protected@edef\@do@subglossentry{\noexpand\glossarysubentryfield{\number#1}%
5485     {#2}%
5486     {\noexpand\glsnamefont
5487       {\expandafter\expandonce\csname glo@#2@name\endcsname}}%
5488     {\expandafter\expandonce\csname glo@#2@desc\endcsname}%
5489     {\expandafter\expandonce\csname glo@#2@symbol\endcsname}%
5490     {\the\toks@}}%
5491   }%
5492   \@do@subglossentry
5493 }

```

sentrycompatibility

```

5494 \newcommand*\setglossentrycompatibility{%
5495   \let\glossentry\compatibleglossentry
5496   \let\subglossentry\compatiblesubglossentry
5497 }
5498 \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.

```

5499 \newcommand{\glossaryentryfield}[5]{%
5500   \GlossariesWarning
5501   {Deprecated use of \string\glossaryentryfield.^^J
5502     I recommend you change to \string\glossentry.^^J
5503     If you've just upgraded, try removing your gls auxiliary
5504     files^^J and recompile}%
5505   \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.)

```
5506 \newcommand*{\glossarysubentryfield}[6]{%
5507   \GlossariesWarning
5508   {Deprecated use of \string\glossarysubentryfield.^^J
5509    I recommend you change to \string\subglossentry.^^J
5510    If you've just upgraded, try removing your gls auxiliary
5511    files^^J and recompile}%
5512   \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`

```
5513 \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`

```
5514 \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{\langle label \rangle}`

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`

```
5515 \newcommand*{\glsgetgrouptitle}[1]{%
5516   \@gls@getgrouptitle{#1}{\@gls@grptitle}%
5517   \@gls@grptitle
5518 }
```

`\@gls@getgrouptitle` Gets the group title specified by the label (first argument) and stores in the second argument, which must be a control sequence.

```
5519 \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.

```
5520 \dtl@ifsingle{#1}%
5521 {%
5522   \ifcsundef{#1groupname}{\def#2{#1}}{\letcs#2{#1groupname}}%
5523 }%
5524 {%
5525   \ifboolexpr{test{\ifstrequal{#1}{glsymbols}}
5526               or test{\ifstrequal{#1}{glsnumbers}}}%
5527   {%
5528     \ifcsundef{#1groupname}{\def#2{#1}}{\letcs#2{#1groupname}}%
5529   }%
5530   {%
5531     \def#2{#1}%
5532   }%
5533 }%
5534 }
```

`@getothergrouptitle` Version for the no-indexing app option:

```
5535 \newcommand*{\@gls@noidx@getgrouptitle}[2]{%
5536   \DTLifint{#1}%
5537   {\edef#2{\char#1\relax}}%
5538   {%
5539     \ifcsundef{#1groupname}{\def#2{#1}}{\letcs#2{#1groupname}}%
5540   }%
5541 }
```

`\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`

```
5542 \newcommand*{\glsgetgrouplabel}[1]{%
5543 \ifthenelse{\equal{#1}{\glssymbolsgroupname}}{\glssymbols}{%
5544 \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`

```
5545 \newcommand*{\setentrycounter}[2][1]{%
5546   \def\@glo@counterprefix{#1}%
5547   \ifx\@glo@counterprefix\@empty
5548     \def\@glo@counterprefix{.}%
5549   \else
5550     \def\@glo@counterprefix{.#1.}%
5551   \fi
5552   \def\glsentrycounter{#2}%
5553 }
```

The current glossary style can be set using `\setglossarystyle{<style>}`.

`\setglossarystyle`

```
5554 \newcommand*{\setglossarystyle}[1]{%
5555   \ifcsundef{@glsstyle@#1}%
5556   {%
5557     \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
5558   }%
5559   {%
5560     \csname @glsstyle@#1\endcsname
5561   }%
5562 }
```

`\glossarystyle`

```
5563 \newcommand*{\glossarystyle}[1]{%
5564   \ifcsundef{@glsstyle@#1}%
5565   {%
5566     \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
5567   }%
5568   {%
5569     \GlossariesWarning
5570     {Deprecated command \string\glossarystyle.^^J
5571     I recommend you switch to \string\setglossarystyle\space unless
5572     you want to maintain backward compatibility}%
5573     \setglossentrycompatibility
5574     \csname @glsstyle@#1\endcsname

5575   \ifcsdef{@glscompstyle@#1}%
5576   {%\setglossentrycompatibility\csuse{@glscompstyle@#1}}%
5577   {}%
```

```
5578 }%
5579 }
```

`\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.

```
5580 \newcommand{\newglossarystyle}[2]{%
5581   \ifcsundef{@glsstyle@#1}%
5582   {%
5583     \expandafter\def\csname @glsstyle@#1\endcsname{#2}%
5584   }%
5585   {%
5586     \PackageError{glossaries}{Glossary style ‘#1’ is already defined}{}%
5587   }%
5588 }
```

`\renewglossarystyle` Code for this macro supplied by Marco Daniel.

```
5589 \newcommand{\renewglossarystyle}[2]{%
5590   \ifcsundef{@glsstyle@#1}%
5591   {%
5592     \PackageError{glossaries}{Glossary style ‘#1’ isn’t already defined}{}%
5593   }%
5594   {%
5595     \csdef{@glsstyle@#1}{#2}%
5596   }%
5597 }
```

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`

```
5598 \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`

```
5599 \ifcsundef{hyperlink}%
5600 {%
5601   \def\glshypernumber#1{#1}%
5602 }%
5603 {%
5604   \def\glshypernumber#1{\@glshypernumber#1\nohyperpage{}}\@nil}
5605 }
```

`\@glshypernumber` This code was provided by Heiko Oberdiek to allow material to be attached to the location.

```
5606 \def\@glshypernumber#1\nohyperpage#2#3\@nil{%
5607   \ifx\#1\%
5608     \else
5609       \@delimR#1\delimR\delimR\%
5610     \fi
5611   \ifx\#2\%
5612     \else
5613       #2%
5614     \fi
5615   \ifx\#3\%
5616     \else
5617       \@glshypernumber#3\@nil
5618     \fi
5619 }
```

`\@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`

```
5620 \def\@delimR#1\delimR #2\delimR #3\%
5621 \ifx\#2\%
5622   \@delimN{#1}%
5623 \else
5624   \@gls@numberlink{#1}\delimR\@gls@numberlink{#2}%
5625 \fi}
```

`\@delimN` displays a list of individual numbers, instead of a range:

\@delimN

```
5626 \def\@delimN#1{\@delimN#1\delimN \delimN\\}%
5627 \def\@delimN#1\delimN #2\delimN#3\\{\%
5628 \ifx\\#3\\%
5629 \@gls@numberlink{#1}%
5630 \else
5631 \@gls@numberlink{#1}\delimN\@gls@numberlink{#2}%
5632 \fi
5633 }
```

The following code is modified from hyperref's \HyInd@pagelink where the name of the counter being used is given by \@gls@counter.

```
5634 \def\@gls@numberlink#1{%
5635 \begingroup
5636 \toks@={}%
5637 \@gls@removespaces#1 \@nil
5638 \endgroup}

5639 \def\@gls@removespaces#1 #2\@nil{%
5640 \toks@=\expandafter{\the\toks@#1}%
5641 \ifx\\#2\\%
5642 \edef\x{\the\toks@}%
5643 \ifx\x\empty
5644 \else

5645 \hyperlink{\glstrycounter\@gls@counterprefix\the\toks@}%
5646 {\the\toks@}%
5647 \fi
5648 \else
5649 \@gls@ReturnAfterFi{%
5650 \@gls@removespaces#2\@nil
5651 }%
5652 \fi
5653 }
5654 \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

```
5655 \newcommand*\hyperrm[1]{\textrm{\glshypernumber{#1}}}
```

\hypersf

```
5656 \newcommand*\hypersf[1]{\textsf{\glshypernumber{#1}}}
```

\hypertt

```
5657 \newcommand*\hypertt[1]{\texttt{\glshypernumber{#1}}}
```



```

\hyperbf
5658 \newcommand*{\hyperbf}[1]{\textbf{\glshypernumber{#1}}}

\hypermd
5659 \newcommand*{\hypermd}[1]{\textmd{\glshypernumber{#1}}}

\hyperit
5660 \newcommand*{\hyperit}[1]{\textit{\glshypernumber{#1}}}

\hypersl
5661 \newcommand*{\hypersl}[1]{\textsl{\glshypernumber{#1}}}

\hyperup
5662 \newcommand*{\hyperup}[1]{\textup{\glshypernumber{#1}}}

\hypersc
5663 \newcommand*{\hypersc}[1]{\textsc{\glshypernumber{#1}}}

\hyperemph
5664 \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.

```

5665 \newcommand{\oldacronym}[4][\gls@label]{%
5666   \def\gls@label{#2}%
5667   \newacronym[#4]{#1}{#2}{#3}%
5668   \ifcsundef{xspace}%
5669     {%

```

```

5670 \expandafter\edef\csname#1\endcsname{%
5671 \noexpand\@ifstar{\noexpand\Gls{#1}}{\noexpand\gls{#1}}}%
5672 }%
5673 }%
5674 {%
5675 \expandafter\edef\csname#1\endcsname{%
5676 \noexpand\@ifstar{\noexpand\Gls{#1}\noexpand\xspace}{%
5677 \noexpand\gls{#1}\noexpand\xspace}%
5678 }%
5679 }%
5680 }

```

`\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`

```

5681 \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 need to cancel it out.

```

5682 \newcommand*{\acrpluralsuffix}{\glspluralsuffix}

```

If `garamondx` has been loaded, need to use `\textulc` instead of `\textup`.

`\glstextup`

```

5683 \newrobustcmd*{\glstextup}[1]{\ifdef\textulc{\textulc{#1}}{\textup{#1}}}

```

The following are defined for compatibility with version 2.07 and earlier.

`\glsshortkey`

```

5684 \newcommand*{\glsshortkey}{short}

```

```

\glsshortpluralkey
5685 \newcommand*{\glsshortpluralkey}{shortplural}

\glslongkey
5686 \newcommand*{\glslongkey}{long}

\glslongpluralkey
5687 \newcommand*{\glslongpluralkey}{longplural}

\acrfull  Full form of the acronym.
5688 \newrobustcmd*{\acrfull}{\@gls@hyp@opt\@ns@acrfull}

5689 \newcommand*\ns@acrfull[2][\%
5690   \new@ifnextchar[\@acrfull{#1}{#2}]{\%
5691     {\@acrfull{#1}{#2}[]}\%
5692 }

\@acrfull  Low-level macro:
5693 \def\@acrfull#1#2[#3]{\%
    Make it easier for acronym styles to change this:
5694   \acrfullfmt{#1}{#2}{#3}\%
5695 }

    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.
5696 \newcommand*{\acrfullfmt}[3]{\%
5697   \acrlinkfullformat{\@acrlong}{\@acrshort}{#1}{#2}{#3}\%
5698 }

\acrlinkfullformat  Format for full links like \acrfull. Syntax: \acrlinkfullformat{<long
cs>}{<short cs>}{<options>}{<label>}{<insert>}
5699 \newcommand{\acrlinkfullformat}[5]{\%
5700   \acrfullformat{#1{#3}{#4}[#5]}{#2{#3}{#4}[]}\%
5701 }

\acrfullformat  Default full form is <long> (<short>).
5702 \newcommand{\acrfullformat}[2]{#1\glsspace(#2)}

\glsspace  Robust space to ensure it's written to the .glsdefs file.
5703 \newrobustcmd{\glsspace}{\space}

    Default format for full acronym

\Acrfull
5704 \newrobustcmd*{\Acrfull}{\@gls@hyp@opt\@ns@Acrfull}

```

```

5705 \newcommand*\ns@Acrfull[2][]{%
5706   \new@ifnextchar[{\@Acrfull{#1}{#2}}%
5707     {\@Acrfull{#1}{#2}[]}%
5708 }

```

Low-level macro:

```

5709 \def\@Acrfull#1#2[#3]{%

```

Make it easier for acronym styles to change this:

```

5710   \Acrfullfmt{#1}{#2}{#3}%
5711 }

```

`\Acrfullfmt` First letter upper case full format.

```

5712 \newcommand*\@Acrfullfmt[3]{%
5713   \acrlinkfullformat{\@Acrlong}{\@acrshort}{#1}{#2}{#3}%
5714 }

```

`\ACRfull`

```

5715 \newrobustcmd*\@ACRfull{\@gls@hyp@opt\ns@ACRfull}

```

```

5716 \newcommand*\ns@ACRfull[2][]{%
5717   \new@ifnextchar[{\@ACRfull{#1}{#2}}%
5718     {\@ACRfull{#1}{#2}[]}%
5719 }

```

Low-level macro:

```

5720 \def\@ACRfull#1#2[#3]{%

```

Make it easier for acronym styles to change this:

```

5721   \ACRfullfmt{#1}{#2}{#3}%
5722 }

```

`\ACRfullfmt` All upper case full format.

```

5723 \newcommand*\@ACRfullfmt[3]{%
5724   \acrlinkfullformat{\@ACRlong}{\@ACRshort}{#1}{#2}{#3}%
5725 }

```

Plural:

`\acrfullpl`

```

5726 \newrobustcmd*\@acrfullpl{\@gls@hyp@opt\ns@acrfullpl}

```

```

5727 \newcommand*\ns@acrfullpl[2][]{%
5728   \new@ifnextchar[{\@acrfullpl{#1}{#2}}%
5729     {\@acrfullpl{#1}{#2}[]}%
5730 }

```

Low-level macro:

```

5731 \def\@acrfullpl#1#2[#3]{%

```

Make it easier for acronym styles to change this:

```
5732 \acrfullplfmt{#1}{#2}{#3}%  
5733 }
```

\acrfullplfmt No case change plural full format.

```
5734 \newcommand*\acrfullplfmt[3]{%  
5735 \acrlinkfullformat{\@acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%  
5736 }
```

\Acrfullpl

```
5737 \newrobustcmd*\Acrfullpl{\@gls@hyp@opt\ns@Acrfullpl}  
  
5738 \newcommand*\ns@Acrfullpl[2][{}]{%  
5739 \new@ifnextchar[{\@Acrfullpl{#1}{#2}}%  
5740 {\@Acrfullpl{#1}{#2}[]}%  
5741 }
```

Low-level macro:

```
5742 \def\@Acrfullpl#1#2[#3]{%
```

Make it easier for acronym styles to change this:

```
5743 \Acrfullplfmt{#1}{#2}{#3}%  
5744 }
```

\Acrfullplfmt First letter upper case plural full format.

```
5745 \newcommand*\Acrfullplfmt[3]{%  
5746 \acrlinkfullformat{\@Acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%  
5747 }
```

\ACRfullpl

```
5748 \newrobustcmd*\ACRfullpl{\@gls@hyp@opt\ns@ACRfullpl}  
  
5749 \newcommand*\ns@ACRfullpl[2][{}]{%  
5750 \new@ifnextchar[{\@ACRfullpl{#1}{#2}}%  
5751 {\@ACRfullpl{#1}{#2}[]}%  
5752 }
```

Low-level macro:

```
5753 \def\@ACRfullpl#1#2[#3]{%
```

Make it easier for acronym styles to change this:

```
5754 \ACRfullplfmt{#1}{#2}{#3}%  
5755 }
```

\ACRfullplfmt All upper case plural full format.

```
5756 \newcommand*\ACRfullplfmt[3]{%  
5757 \acrlinkfullformat{\@ACRlongpl}{\@ACRshortpl}{#1}{#2}{#3}%  
5758 }
```

1.17 Predefined acronym styles

`\acronymfont` This is only used with the additional acronym styles:

```
5759 \newcommand{\acronymfont}[1]{#1}
```

`\firstacronymfont` This is only used with the additional acronym styles:

```
5760 \newcommand{\firstacronymfont}[1]{\acronymfont{#1}}
```

`\acrnameformat` The styles that allow an additional description use `\acrnameformat{<short>}{<long>}` to determine what information is displayed in the name.

```
5761 \newcommand*{\acrnameformat}[2]{\acronymfont{#1}}
```

Define some tokens used by `\newacronym`:

`\glskeylisttok`

```
5762 \newtoks\glskeylisttok
```

`\glslabeltok`

```
5763 \newtoks\glslabeltok
```

`\glsshorttok`

```
5764 \newtoks\glsshorttok
```

`\glslongtok`

```
5765 \newtoks\glslongtok
```

`\newacronymhook` Provide a hook for `\newacronym`:

```
5766 \newcommand*{\newacronymhook}{}
```

`\SetGenericNewAcronym` New improved version of setting the acronym style.

```
5767 \newcommand*{\SetGenericNewAcronym}{%
```

Change the behaviour of `\Glsentryname` to workaround expansion issues that cause a problem for `\makefirstuc`

```
5768 \let\@Gls@entryname\@Gls@acentryname
```

Change the way acronyms are defined:

```
5769 \renewcommand{\newacronym}[4][[]]{%
5770 \ifdefempty{\@glsacronymlists}%
5771 {%
5772 \def\@glo@type{\acronymtype}%
5773 \setkeys{glossentry}{##1}%
5774 \DeclareAcronymList{\@glo@type}%
5775 }%
5776 }%
5777 \glskeylisttok{##1}%
5778 \glslabeltok{##2}%
5779 \glsshorttok{##3}%
5780 \glslongtok{##4}%
```

```

5781 \newacronymhook
5782 \protected@edef\@do@newglossaryentry{%
5783 \noexpand\newglossaryentry{\the\glslabeltok}%
5784 {%
5785 type=\acronymtype,%
5786 name={\expandonce{\acronymentry{##2}}},%
5787 sort={\acronymsort{\the\glsshorttok}{\the\glslongtok}},%
5788 text={\the\glsshorttok},%
5789 short={\the\glsshorttok},%
5790 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5791 long={\the\glslongtok},%
5792 longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5793 \GenericAcronymFields,%
5794 \the\glskeylisttok
5795 }%
5796 }%
5797 \@do@newglossaryentry
5798 }%

```

Make sure that \acrfull etc reflects the new style:

```

5799 \renewcommand*\acrfullfmt}[3]{%
5800 \glslink[##1]{##2}{\genacrfullformat{##2}{##3}}}%
5801 \renewcommand*\Acrfullfmt}[3]{%
5802 \glslink[##1]{##2}{\Genacrfullformat{##2}{##3}}}%
5803 \renewcommand*\ACRfullfmt}[3]{%
5804 \glslink[##1]{##2}{%
5805 \mfirstucMakeUppercase{\genacrfullformat{##2}{##3}}}}%
5806 \renewcommand*\acrfullplfmt}[3]{%
5807 \glslink[##1]{##2}{\genplacrfullformat{##2}{##3}}}%
5808 \renewcommand*\Acrfullplfmt}[3]{%
5809 \glslink[##1]{##2}{\Genplacrfullformat{##2}{##3}}}%
5810 \renewcommand*\ACRfullplfmt}[3]{%
5811 \glslink[##1]{##2}{%
5812 \mfirstucMakeUppercase{\genplacrfullformat{##2}{##3}}}}%

```

Make sure that \glsentryfull etc reflects the new style:

```

5813 \renewcommand*\glsentryfull}[1]{\genacrfullformat{##1}{}}%
5814 \renewcommand*\Glsentryfull}[1]{\Genacrfullformat{##1}{}}%
5815 \renewcommand*\glsentryfullpl}[1]{\genplacrfullformat{##1}{}}%
5816 \renewcommand*\Glsentryfullpl}[1]{\Genplacrfullformat{##1}{}}%
5817 }

```

`\GenericAcronymFields` Fields used by \SetGenericNewAcronym that can be changed by the acronym style.

```

5818 \newcommand*\GenericAcronymFields{description={\the\glslongtok}}

```

`\acronymentry` `\acronymentry{<label>}`

Display style for the name field in the list of acronyms.

```
5819 \newcommand*{\acronymentry}[1]{\acronymfont{\glsentryshort{#1}}}
```

`\acronymsort` `\acronymsort{<short>}{<long>}`

Default sort format for acronyms.

```
5820 \newcommand*{\acronymsort}[2]{#1}
```

`\setacronymstyle` `\setacronymstyle{<style name>}`

```
5821 \newcommand*{\setacronymstyle}[1]{%
5822   \ifcsundef{@glsacr@dispstyle@#1}
5823   {%
5824     \PackageError{glossaries}{Undefined acronym style ‘#1’}{%
5825     }%
5826   {%
5827     \ifdefempty{@glsacronymlists}%
5828     {%
5829       \DeclareAcronymList{\acronymtype}%
5830     }%
5831     {}%
5832     \SetGenericNewAcronym
5833     \GlsUseAcrStyleDefs{#1}%
5834     \@for\@gls@type:=\@glsacronymlists\do{%
5835       \defglsentryfmt[\@gls@type]{\GlsUseAcrEntryDispStyle{#1}}%
5836     }%
5837   }%
5838 }
```

`\newacronymstyle` `\newacronymstyle{<style name>}{<entry format definition>}{<display definitions>}`

Defines a new acronym style called *<style name>*.

```
5839 \newcommand*{\newacronymstyle}[3]{%
5840   \ifcsdef{@glsacr@dispstyle@#1}%
5841   {%
5842     \PackageError{glossaries}{Acronym style ‘#1’ already exists}{}%
5843   }%
5844   {%
5845     \csdef{@glsacr@dispstyle@#1}{#2}%
5846     \csdef{@glsacr@styledefs@#1}{#3}%
5847   }%
5848 }
```

`\renewacronymstyle` Redefines the given acronym style.


```

5849 \newcommand*{\renewacronymstyle}[3]{%
5850   \ifcsdef{@glsacr@dispstyle@#1}%
5851   {%
5852     \csdef{@glsacr@dispstyle@#1}{#2}%
5853     \csdef{@glsacr@styledefs@#1}{#3}%
5854   }%
5855   {%
5856     \PackageError{glossaries}{Acronym style ‘#1’ doesn’t exist}{}%
5857   }%
5858 }

```

seAcrEntryDispStyle

```

5859 \newcommand*{\GlsUseAcrEntryDispStyle}[1]{\csuse{@glsacr@dispstyle@#1}}

```

\GlsUseAcrStyleDefs

```

5860 \newcommand*{\GlsUseAcrStyleDefs}[1]{\csuse{@glsacr@styledefs@#1}}

```

Predefined acronym styles:

long-short *(long)* (*short*) acronym style.

```

5861 \newacronymstyle{long-short}%
5862 {%

```

Check for long form in case this is a mixed glossary.

```

5863   \ifglshaslong{\glslabel}{\glsgenacfmt}{\glsgenentryfmt}%
5864 }%
5865 {%
5866   \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
5867   \renewcommand*{\genacrfullformat}[2]{%
5868     \glsentrylong{##1}##2\space
5869     (\protect\firstacronymfont{\glsentryshort{##1}})}%
5870   }%
5871   \renewcommand*{\Genacrfullformat}[2]{%
5872     \Glsentrylong{##1}##2\space
5873     (\protect\firstacronymfont{\glsentryshort{##1}})}%
5874   }%
5875   \renewcommand*{\genplacrfullformat}[2]{%
5876     \glsentrylongpl{##1}##2\space
5877     (\protect\firstacronymfont{\glsentryshortpl{##1}})}%
5878   }%
5879   \renewcommand*{\Genplacrfullformat}[2]{%
5880     \Glsentrylongpl{##1}##2\space
5881     (\protect\firstacronymfont{\glsentryshortpl{##1}})}%
5882   }%
5883   \renewcommand*{\acronymentry}[1]{\acronymfont{\glsentryshort{##1}}}
5884   \renewcommand*{\acronymsort}[2]{##1}%
5885   \renewcommand*{\acronymfont}[1]{##1}%
5886   \renewcommand*{\firstacronymfont}[1]{\acronymfont{##1}}%
5887   \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
5888 }

```

short-long *<short>* (*<long>*) acronym style.

```
5889 \newacronymstyle{short-long}%
5890 {%
    Check for long form in case this is a mixed glossary.
5891   \ifglshaslong{\glslabel}{\glsgenacfmt}{\glsgenentryfmt}%
5892 }%
5893 {%
5894   \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
5895   \renewcommand*{\genacrfullformat}[2]{%
5896     \protect\firstacronymfont{\glsentryshort{##1}}##2\space
5897     (\glsentrylong{##1})%
5898   }%
5899   \renewcommand*{\Genacrfullformat}[2]{%
5900     \protect\firstacronymfont{\Glsentryshort{##1}}##2\space
5901     (\glsentrylong{##1})%
5902   }%
5903   \renewcommand*{\genplacrfullformat}[2]{%
5904     \protect\firstacronymfont{\glsentryshortpl{##1}}##2\space
5905     (\glsentrylongpl{##1})%
5906   }%
5907   \renewcommand*{\Genplacrfullformat}[2]{%
5908     \protect\firstacronymfont{\Glsentryshortpl{##1}}##2\space
5909     (\glsentrylongpl{##1})%
5910   }%
5911   \renewcommand*{\acronymentry}[1]{\acronymfont{\glsentryshort{##1}}}%
5912   \renewcommand*{\acronymsort}[2]{##1}%
5913   \renewcommand*{\acronymfont}[1]{##1}%
5914   \renewcommand*{\firstacronymfont}[1]{\acronymfont{##1}}%
5915   \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
5916 }
```

long-sc-short *<long>* (\textsc{<short>}) acronym style.

```
5917 \newacronymstyle{long-sc-short}%
5918 {%
5919   \GlsUseAcrEntryDisplayStyle{long-short}%
5920 }%
5921 {%
5922   \GlsUseAcrStyleDefs{long-short}%
5923   \renewcommand{\acronymfont}[1]{\textsc{##1}}%
5924   \renewcommand*{\acrpluralsuffix}{\glstextup{\glspluralsuffix}}%
5925 }
```

long-sm-short *<long>* (\textsmaller{<short>}) acronym style.

```
5926 \newacronymstyle{long-sm-short}%
5927 {%
5928   \GlsUseAcrEntryDisplayStyle{long-short}%
5929 }%
5930 {%
```

```

5931 \GlsUseAcrStyleDefs{long-short}%
5932 \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
5933 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
5934 }

```

sc-short-long $\langle short \rangle$ ($\textsc{\langle long \rangle}$) acronym style.

```

5935 \newacronymstyle{sc-short-long}%
5936 {%
5937 \GlsUseAcrEntryDisplayStyle{short-long}%
5938 }%
5939 {%
5940 \GlsUseAcrStyleDefs{short-long}%
5941 \renewcommand{\acronymfont}[1]{\textsc{##1}}%
5942 \renewcommand*{\acrpluralsuffix}{\glstextup{\glspluralsuffix}}%
5943 }

```

sm-short-long $\langle short \rangle$ ($\textsmaller{\langle long \rangle}$) acronym style.

```

5944 \newacronymstyle{sm-short-long}%
5945 {%
5946 \GlsUseAcrEntryDisplayStyle{short-long}%
5947 }%
5948 {%
5949 \GlsUseAcrStyleDefs{short-long}%
5950 \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
5951 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
5952 }

```

long-short-desc $\langle long \rangle$ ($\langle short \rangle$) acronym style that has an accompanying description (which the user needs to supply).

```

5953 \newacronymstyle{long-short-desc}%
5954 {%
5955 \GlsUseAcrEntryDisplayStyle{long-short}%
5956 }%
5957 {%
5958 \GlsUseAcrStyleDefs{long-short}%
5959 \renewcommand*{\GenericAcronymFields}{}%
5960 \renewcommand*{\acronymsort}[2]{##2}%
5961 \renewcommand*{\acronymentry}[1]{%
5962 \glsentrylong{##1}\space (\acronymfont{\glsentryshort{##1}})}%
5963 }

```

long-sc-short-desc $\langle long \rangle$ ($\textsc{\langle short \rangle}$) acronym style that has an accompanying description (which the user needs to supply).

```

5964 \newacronymstyle{long-sc-short-desc}%
5965 {%
5966 \GlsUseAcrEntryDisplayStyle{long-sc-short}%
5967 }%
5968 {%

```

```

5969 \GlsUseAcrStyleDefs{long-sm-short}%
5970 \renewcommand*{\GenericAcronymFields}{}%
5971 \renewcommand*{\acronymsort}[2]{##2}%
5972 \renewcommand*{\acronymentry}[1]{%
5973     \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
5974 }

```

long-sm-short-desc *⟨long⟩* (\textsmaller{⟨short⟩}) acronym style that has an accompanying description (which the user needs to supply).

```

5975 \newacronymstyle{long-sm-short-desc}%
5976 {%
5977     \GlsUseAcrEntryDisplayStyle{long-sm-short}%
5978 }%
5979 {%
5980     \GlsUseAcrStyleDefs{long-sm-short}%
5981     \renewcommand*{\GenericAcronymFields}{}%
5982     \renewcommand*{\acronymsort}[2]{##2}%
5983     \renewcommand*{\acronymentry}[1]{%
5984         \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
5985 }

```

short-long-desc *⟨short⟩* ({⟨long⟩}) acronym style that has an accompanying description (which the user needs to supply).

```

5986 \newacronymstyle{short-long-desc}%
5987 {%
5988     \GlsUseAcrEntryDisplayStyle{short-long}%
5989 }%
5990 {%
5991     \GlsUseAcrStyleDefs{short-long}%
5992     \renewcommand*{\GenericAcronymFields}{}%
5993     \renewcommand*{\acronymsort}[2]{##2}%
5994     \renewcommand*{\acronymentry}[1]{%
5995         \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
5996 }

```

sc-short-long-desc *⟨long⟩* (\textsc{⟨short⟩}) acronym style that has an accompanying description (which the user needs to supply).

```

5997 \newacronymstyle{sc-short-long-desc}%
5998 {%
5999     \GlsUseAcrEntryDisplayStyle{sc-short-long}%
6000 }%
6001 {%
6002     \GlsUseAcrStyleDefs{sc-short-long}%
6003     \renewcommand*{\GenericAcronymFields}{}%
6004     \renewcommand*{\acronymsort}[2]{##2}%
6005     \renewcommand*{\acronymentry}[1]{%
6006         \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
6007 }

```

sm-short-long-desc *<long>* (\textsmaller{<short>}) acronym style that has an accompanying description (which the user needs to supply).

```

6008 \newacronymstyle{sm-short-long-desc}%
6009 {%
6010   \GlsUseAcrEntryDispStyle{sm-short-long}%
6011 }%
6012 {%
6013   \GlsUseAcrStyleDefs{sm-short-long}%
6014   \renewcommand*{\GenericAcronymFields}{}%
6015   \renewcommand*{\acronymsort}[2]{##2}%
6016   \renewcommand*{\acronymentry}[1]{%
6017     \glsentrylong{##1}\space (\acronymfont{\glsentryshort{##1}})%
6018 }

```

dua *<long>* only acronym style.

```

6019 \newacronymstyle{dua}%
6020 {%

```

Check for long form in case this is a mixed glossary.

```

6021   \ifdefempty\glscustomtext
6022   {%
6023     \ifglshaslong{\glslabel}%
6024     {%
6025       \glsifplural
6026       {%

```

Plural form:

```

6027         \glscapscase
6028         {%

```

Plural form, don't adjust case:

```

6029         \glsentrylongpl{\glslabel}\glsinsert
6030         }%
6031         {%

```

Plural form, make first letter upper case:

```

6032         \Glsentrylongpl{\glslabel}\glsinsert
6033         }%
6034         {%

```

Plural form, all caps:

```

6035         \mfirstucMakeUppercase
6036         {\glsentrylongpl{\glslabel}\glsinsert}%
6037         }%
6038         }%
6039         {%

```

Singular form

```

6040         \glscapscase
6041         {%

```

Singular form, don't adjust case:

```
6042      \glsentrylong{\glslabel}\glsinsert
6043      }%
6044      {%
```

Subsequent singular form, make first letter upper case:

```
6045      \Glsentrylong{\glslabel}\glsinsert
6046      }%
6047      {%
```

Subsequent singular form, all caps:

```
6048      \mfirstucMakeUppercase
6049      {\glsentrylong{\glslabel}\glsinsert}%
6050      }%
6051      }%
6052      }%
6053      {%
```

Not an acronym:

```
6054      \glsgenentryfmt
6055      }%
6056      }%
6057      {\glscustomtext\glsinsert}%
6058      }%
6059      {%
6060      \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%

6061      \renewcommand*{\acrfullfmt}[3]{%
6062          \glslink[##1]{##2}{\glsentrylong{##2}##3\space
6063              (\acronymfont{\glsentryshort{##2}})}}%
6064      \renewcommand*{\Acrfullfmt}[3]{%
6065          \glslink[##1]{##2}{\Glsentrylong{##2}##3\space
6066              (\acronymfont{\glsentryshort{##2}})}}%
6067      \renewcommand*{\ACRfullfmt}[3]{%
6068          \glslink[##1]{##2}{%
6069              \mfirstucMakeUppercase{\glsentrylong{##2}##3\space
6070              (\acronymfont{\glsentryshort{##2}})}}}%

6071      \renewcommand*{\acrfullplfmt}[3]{%
6072          \glslink[##1]{##2}{\glsentrylongpl{##2}##3\space
6073              (\acronymfont{\glsentryshortpl{##2}})}}%

6074      \renewcommand*{\Acrfullplfmt}[3]{%
6075          \glslink[##1]{##2}{\Glsentrylongpl{##2}##3\space
6076              (\acronymfont{\glsentryshortpl{##2}})}}%
6077      \renewcommand*{\ACRfullplfmt}[3]{%
6078          \glslink[##1]{##2}{%
6079              \mfirstucMakeUppercase{\glsentrylongpl{##2}##3\space
6080              (\acronymfont{\glsentryshortpl{##2}})}}}%
6081      \renewcommand*{\glsentryfull}[1]{%
6082          \glsentrylong{##1}\space(\acronymfont{\glsentryshort{##1}})%
```

```

6083 }%
6084 \renewcommand*{\Glsentryfull}[1]{%
6085   \Glsentrylong{##1}\space(\acronymfont{\Glsentryshort{##1}})%
6086 }%
6087 \renewcommand*{\Glsentryfullpl}[1]{%
6088   \Glsentrylongpl{##1}\space(\acronymfont{\Glsentryshortpl{##1}})%
6089 }%
6090 \renewcommand*{\Glsentryfullpl}[1]{%
6091   \Glsentrylongpl{##1}\space(\acronymfont{\Glsentryshortpl{##1}})%
6092 }%
6093 \renewcommand*{\acronymentry}[1]{\acronymfont{\Glsentryshort{##1}}}%
6094 \renewcommand*{\acronymsort}[2]{##1}%
6095 \renewcommand*{\acronymfont}[1]{##1}%
6096 \renewcommand*{\acrpluralsuffix}{\Glspluralsuffix}%
6097 }

```

dua-desc *<long>* only acronym style with user-supplied description.

```

6098 \newacronymstyle{dua-desc}%
6099 {%
6100   \GlsUseAcrEntryDispStyle{dua}%
6101 }%
6102 {%
6103   \GlsUseAcrStyleDefs{dua}%
6104   \renewcommand*{\GenericAcronymFields}{}%
6105   \renewcommand*{\acronymentry}[1]{\acronymfont{\Glsentrylong{##1}}}%
6106   \renewcommand*{\acronymsort}[2]{##2}%
6107 }%

```

footnote *<short>*\footnote{*<long>*} acronym style.

```

6108 \newacronymstyle{footnote}%
6109 {%
6110   \ifglshaslong{\Glslabel}{\GlsGenacFmt}{\GlsGenentryFmt}%
6111 }%
6112 {%
6113   \renewcommand*{\GenericAcronymFields}{description={\the\GlsLongTok}}%
6114   \GlsHyperfirstfalse
6115   \renewcommand*{\GenacrFullFormat}[2]{%
6116     \protect\firstacronymfont{\Glsentryshort{##1}}##2%
6117     \protect\footnote{\Glsentrylong{##1}}%
6118   }%
6119   \renewcommand*{\GenacrFullFormat}[2]{%
6120     \firstacronymfont{\Glsentryshort{##1}}##2%
6121     \protect\footnote{\Glsentrylong{##1}}%
6122   }%
6123   \renewcommand*{\GenplacrFullFormat}[2]{%

```

```

6124 \protect\firstacronymfont{\glentryshortpl{##1}}##2%
6125 \protect\footnote{\glentrylongpl{##1}}%
6126 }%
6127 \renewcommand*{\Genplacrfullformat}[2]{%
6128 \protect\firstacronymfont{\glentryshortpl{##1}}##2%
6129 \protect\footnote{\glentrylongpl{##1}}%
6130 }%
6131 \renewcommand*{\acronymentry}[1]{\acronymfont{\glentryshort{##1}}}%
6132 \renewcommand*{\acronymsort}[2]{##1}%
6133 \renewcommand*{\acronymfont}[1]{##1}%
6134 \renewcommand*{\acrpluralsuffix}{\glpluralsuffix}%

```

Don't use footnotes for \acrfull:

```

6135 \renewcommand*{\acrfullfmt}[3]{%
6136 \glslink[##1]{##2}{\acronymfont{\glentryshort{##2}}##3\space
6137 (\glentrylong{##2})}%
6138 \renewcommand*{\Acrfullfmt}[3]{%
6139 \glslink[##1]{##2}{\acronymfont{\glentryshort{##2}}##3\space
6140 (\glentrylong{##2})}%
6141 \renewcommand*{\ACRfullfmt}[3]{%
6142 \glslink[##1]{##2}{%
6143 \mfirstucMakeUppercase{\acronymfont{\glentryshort{##2}}##3\space
6144 (\glentrylong{##2})}}}%
6145 \renewcommand*{\acrfullplfmt}[3]{%
6146 \glslink[##1]{##2}{\acronymfont{\glentryshortpl{##2}}##3\space
6147 (\glentrylongpl{##2})}%
6148 \renewcommand*{\Acrfullplfmt}[3]{%
6149 \glslink[##1]{##2}{\acronymfont{\glentryshortpl{##2}}##3\space
6150 (\glentrylongpl{##2})}%
6151 \renewcommand*{\ACRfullplfmt}[3]{%
6152 \glslink[##1]{##2}{%
6153 \mfirstucMakeUppercase{\acronymfont{\glentryshortpl{##2}}##3\space
6154 (\glentrylongpl{##2})}}}%

```

Similarly for \glentryfull etc:

```

6155 \renewcommand*{\glentryfull}[1]{%
6156 \acronymfont{\glentryshort{##1}}\space(\glentrylong{##1})}%
6157 \renewcommand*{\Glsentryfull}[1]{%
6158 \acronymfont{\Glsentryshort{##1}}\space(\glentrylong{##1})}%
6159 \renewcommand*{\glentryfullpl}[1]{%
6160 \acronymfont{\glentryshortpl{##1}}\space(\glentrylongpl{##1})}%
6161 \renewcommand*{\Glsentryfullpl}[1]{%
6162 \acronymfont{\Glsentryshortpl{##1}}\space(\glentrylongpl{##1})}%
6163 }

```

footnote-sc \textsc{<short>}\footnote{<long>} acronym style.

```

6164 \newacronymstyle{footnote-sc}%
6165 {%
6166 \GlsUseAcrEntryDispStyle{footnote}%
6167 }%

```



```

6168 {%
6169   \GlsUseAcrStyleDefs{footnote}%
6170   \renewcommand{\acronymentry}[1]{\acronymfont{\glsentryshort{##1}}}
6171   \renewcommand{\acronymfont}[1]{\textsc{##1}}%
6172   \renewcommand*\{acrpluralsuffix}{\glstextup{\glspluralsuffix}}%
6173 }%

```

footnote-sm \textsmaller{<short>}\footnote{<long>} acronym style.

```

6174 \newacronymstyle{footnote-sm}%
6175 {%
6176   \GlsUseAcrEntryDispStyle{footnote}%
6177 }%
6178 {%
6179   \GlsUseAcrStyleDefs{footnote}%
6180   \renewcommand{\acronymentry}[1]{\acronymfont{\glsentryshort{##1}}}
6181   \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
6182   \renewcommand*\{acrpluralsuffix}{\glspluralsuffix}%
6183 }%

```

footnote-desc <short>\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```

6184 \newacronymstyle{footnote-desc}%
6185 {%
6186   \GlsUseAcrEntryDispStyle{footnote}%
6187 }%
6188 {%
6189   \GlsUseAcrStyleDefs{footnote}%
6190   \renewcommand*\{GenericAcronymFields}{}%
6191   \renewcommand*\{acronymsort}[2]{##2}%
6192   \renewcommand*\{acronymentry}[1]{%
6193     \glsentrylong{##1}\space (\acronymfont{\glsentryshort{##1}})}%
6194 }

```

footnote-sc-desc \textsc{<short>}\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```

6195 \newacronymstyle{footnote-sc-desc}%
6196 {%
6197   \GlsUseAcrEntryDispStyle{footnote-sc}%
6198 }%
6199 {%
6200   \GlsUseAcrStyleDefs{footnote-sc}%
6201   \renewcommand*\{GenericAcronymFields}{}%
6202   \renewcommand*\{acronymsort}[2]{##2}%
6203   \renewcommand*\{acronymentry}[1]{%
6204     \glsentrylong{##1}\space (\acronymfont{\glsentryshort{##1}})}%
6205 }

```

footnote-sm-desc \textsmaller{<short>}\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```

6206 \newacronymstyle{footnote-sm-desc}%
6207 {%
6208   \GlsUseAcrEntryDispStyle{footnote-sm}%
6209 }%
6210 {%
6211   \GlsUseAcrStyleDefs{footnote-sm}%
6212   \renewcommand*{\GenericAcronymFields}{}%
6213   \renewcommand*{\acronymsort}[2]{##2}%
6214   \renewcommand*{\acronymentry}[1]{%
6215     \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
6216 }

```

fineAcronymSynonyms

```

6217 \newcommand*{\DefineAcronymSynonyms}{%

```

Short form

\acs

```

6218   \let\acs\acrshort

```

First letter uppercase short form

\Acs

```

6219   \let\Acs\Acrshort

```

Plural short form

\acsp

```

6220   \let\acsp\acrshortpl

```

First letter uppercase plural short form

\Acsp

```

6221   \let\Acsp\Acrshortpl

```

Long form

\acl

```

6222   \let\acl\aclong

```

Plural long form

\aclp

```

6223   \let\aclp\aclongpl

```

First letter upper case long form

\Acl

```

6224   \let\Acl\Aclong

```

First letter upper case plural long form

```

\Aclp
6225 \let\Aclp\Acrlongpl

Full form

\acf
6226 \let\acf\acrfull

Plural full form

\acfp
6227 \let\acfp\acrfullpl

First letter upper case full form

\Acf
6228 \let\Acf\Acrfull

First letter upper case plural full form

\Acfp
6229 \let\Acfp\Acrfullpl

Standard form

\ac
6230 \let\ac\gls

First upper case standard form

\Ac
6231 \let\Ac\Gls

Standard plural form

\acp
6232 \let\acp\glspl

Standard first letter upper case plural form

\Acp
6233 \let\Acp\Glspl
6234 }

Define synonyms if required
6235 \ifglsacrshortcuts
6236 \DefineAcronymSynonyms
6237 \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.

```
6238 \newcommand*{\SetDefaultAcronymDisplayStyle}[1]{%
6239   \def\glsentryfmt[#1]{\glsentryfmt}%
6240 }
```

defaultNewAcronymDef Sets up the acronym definition for the default style. The information is provided by the tokens `\glslabeltok`, `\glsshorttok`, `\glslongtok` and `\glskeylisttok`.

```
6241 \newcommand*{\DefaultNewAcronymDef}{%
6242   \edef\@do@newglossaryentry{%
6243     \noexpand\newglossaryentry{\the\glslabeltok}%
6244     {%
6245       type=\acronymtype,%
6246       name={\the\glsshorttok},%
6247       sort={\the\glsshorttok},%
6248       text={\the\glsshorttok},%
6249       first={\acrfullformat{\the\glslongtok}{\the\glsshorttok}},%
6250       plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6251       firstplural={\acrfullformat{\noexpand\expandonce\noexpand\@glo@longpl}%
6252                     {\noexpand\expandonce\noexpand\@glo@shortpl}},%
6253       short={\the\glsshorttok},%
6254       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6255       long={\the\glslongtok},%
6256       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6257       description={\the\glslongtok},%
6258       descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
```

Remaining options specified by the user:

```
6259     \the\glskeylisttok
6260   }%
6261 }%
6262 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6263 \let\@org@gls@assign@plural\gls@assign@plural
6264 \let\@org@gls@assign@descplural\gls@assign@descplural
6265 \def\gls@assign@firstpl##1##2{%
6266   \@@gls@expand@field{##1}{firstpl}{##2}%
6267 }%
6268 \def\gls@assign@plural##1##2{%
6269   \@@gls@expand@field{##1}{plural}{##2}%
6270 }%
6271 \def\gls@assign@descplural##1##2{%
6272   \@@gls@expand@field{##1}{descplural}{##2}%
6273 }%
6274 \@do@newglossaryentry
6275 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6276 \let\gls@assign@plural\@org@gls@assign@plural
6277 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6278 }
```

DefaultAcronymStyle Set up the default acronym style:

```
6279 \newcommand*\SetDefaultAcronymStyle{%
```

Set the display style:

```
6280 \@for\@gls@type:=\@glsacronymlists\do{%
6281     \SetDefaultAcronymDisplayStyle{\@gls@type}%
6282 }%
```

Set up the definition of \newacronym:

```
6283 \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).

```
6284     \ifx\@glsacronymlists\@empty
6285         \def\@glo@type{\acronymtype}%
6286         \setkeys{glossentry}{##1}%
6287         \DeclareAcronymList{\@glo@type}%
6288         \SetDefaultAcronymDisplayStyle{\@glo@type}%
6289     \fi
6290     \glskeylisttok{##1}%
6291     \glslabeltok{##2}%
6292     \glsshorttok{##3}%
6293     \glslongtok{##4}%
6294     \newacronymhook
6295     \DefaultNewAcronymDef
6296 }%
6297 \renewcommand*\acrpluralsuffix{\glspluralsuffix}%
6298 }
```

\acrfootnote Used by the footnote acronym styles.

```
6299 \newcommand*\acrfootnote}[3]{\acrlinkfootnote{#1}{#2}{#3}}
```

\acrlinkfootnote

```
6300 \newcommand*\acrlinkfootnote}[3]{%
6301     \footnote{\glslink[#1]{#2}{#3}}%
6302 }
```

\acrnoflinkfootnote

```
6303 \newcommand*\acrnoflinkfootnote}[3]{%
6304     \footnote{#3}%
6305 }
```

AcronymDisplayStyle Sets the acronym display style for given glossary for the description and footnote combination.

```
6306 \newcommand*\SetDescriptionFootnoteAcronymDisplayStyle}[1]{%
6307     \def\glsentryfmt[#1]{%
```

```

6308 \ifdefempty\glscustomtext
6309 {%
6310 \ifglssused{\glslabel}%
6311 {%
6312 \acronymfont{\glsgenentryfmt}%
6313 }%
6314 {%
6315 \firstacronymfont{\glsgenentryfmt}%
6316 \ifglsshassymbol{\glslabel}%
6317 {%
6318 \expandafter\protect\expandafter\acrfootnote\expandafter
6319 {\@gls@link@opts}{\@gls@link@label}%
6320 {%
6321 \glsifplural
6322 {\glsentrysymbolplural{\glslabel}}%
6323 {\glsentrysymbol{\glslabel}}%
6324 }%
6325 }%
6326 }%
6327 }%
6328 {\glscustomtext\glsinsert}%
6329 }%
6330 }

```

otnoteNewAcronymDef

```

6331 \newcommand*{\DescriptionFootnoteNewAcronymDef}{%
6332 \edef\@do@newglossaryentry{%
6333 \noexpand\newglossaryentry{\the\glslabelltok}%
6334 {%
6335 type=\acronymtype,%
6336 name={\noexpand\acronymfont{\the\glsshorttok}},%
6337 sort={\the\glsshorttok},%
6338 first={\the\glsshorttok},%
6339 firstplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6340 text={\the\glsshorttok},%
6341 plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6342 short={\the\glsshorttok},%
6343 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6344 long={\the\glslongtok},%
6345 longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6346 symbol={\the\glslongtok},%
6347 symbolplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6348 \the\glskeylisttok
6349 }%
6350 }%
6351 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6352 \let\@org@gls@assign@plural\gls@assign@plural
6353 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6354 \def\gls@assign@firstpl##1##2{%

```

```

6355 \@@gls@expand@field{##1}{firstpl}{##2}%
6356 }%
6357 \def\gls@assign@plural##1##2{%
6358 \@@gls@expand@field{##1}{plural}{##2}%
6359 }%
6360 \def\gls@assign@symbolplural##1##2{%
6361 \@@gls@expand@field{##1}{symbolplural}{##2}%
6362 }%
6363 \do@newglossaryentry
6364 \let\gls@assign@plural\@org@gls@assign@plural
6365 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6366 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6367 }

```

footnoteAcronymStyle 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.

```

6368 \newcommand*\SetDescriptionFootnoteAcronymStyle{%
6369 \renewcommand{\newacronym}[4][\]{%
6370 \ifx\@glsacronymlists\@empty
6371 \def\@glo@type{\acronymtype}%
6372 \setkeys{glossentry}{##1}%
6373 \DeclareAcronymList{\@glo@type}%
6374 \SetDescriptionFootnoteAcronymDisplayStyle{\@glo@type}%
6375 \fi
6376 \glskeylisttok{##1}%
6377 \glslabeltok{##2}%
6378 \glsshorttok{##3}%
6379 \glslongtok{##4}%
6380 \newacronymhook
6381 \DescriptionFootnoteNewAcronymDef
6382 }%

```

If footnote package option is specified, set the first use to append the long form (stored in symbol) as a footnote.

```

6383 \@for\@gls@type:=\@glsacronymlists\do{%
6384 \SetDescriptionFootnoteAcronymDisplayStyle{\@gls@type}%
6385 }%

```

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.

```

6386 \ifglsacrsmallcaps
6387 \renewcommand*\acronymfont{[1]{\textsc{##1}}}%
6388 \renewcommand*\acrpluralsuffix{%
6389 \glstextup{\glspluralsuffix}}%
6390 \else
6391 \ifglsacrsmaller

```

```

6392     \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%
6393     \fi
6394     \fi

```

Check for package option clash

```

6395     \ifglssacrdua
6396     \PackageError{glossaries}{Option clash: ‘footnote’ and ‘dua’
6397     can’t both be set}{}%
6398     \fi
6399 }%

```

AcronymDisplayStyle Sets the acronym display style for given glossary with description and dua combination.

```

6400 \newcommand*{\SetDescriptionDUAAcronymDisplayStyle}[1]{%
6401     \defglsentryfmt[1]{\glsgenentryfmt}%
6402 }

```

ionDUANewAcronymDef

```

6403 \newcommand*{\DescriptionDUANewAcronymDef}{%
6404     \edef\@do@newglossaryentry{%
6405         \noexpand\newglossaryentry{\the\glslabeltok}%
6406         {%
6407             type=\acronymtype,%
6408             name={\the\glslongtok},%
6409             sort={\the\glslongtok},%
6410             text={\the\glslongtok},%
6411             first={\the\glslongtok},%
6412             plural={\noexpand\expandonce\noexpand\@glo@longpl},%
6413             firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6414             short={\the\glsshorttok},%
6415             shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6416             long={\the\glslongtok},%
6417             longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6418             symbol={\the\glsshorttok},%
6419             symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6420             \the\glskeylisttok
6421         }%
6422     }%
6423     \let\@org@gls@assign@firstpl\gls@assign@firstpl
6424     \let\@org@gls@assign@plural\gls@assign@plural
6425     \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6426     \def\gls@assign@firstpl##1##2{%
6427         \@@gls@expand@field{##1}{firstpl}{##2}%
6428     }%
6429     \def\gls@assign@plural##1##2{%
6430         \@@gls@expand@field{##1}{plural}{##2}%
6431     }%
6432     \def\gls@assign@symbolplural##1##2{%
6433         \@@gls@expand@field{##1}{symbolplural}{##2}%

```



```

6434 }%
6435 \@do@newglossaryentry
6436 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6437 \let\gls@assign@plural\@org@gls@assign@plural
6438 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6439 }

```

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.

```

6440 \newcommand*\SetDescriptionDUAAcronymStyle{%
6441   \ifglsacrsmallcaps
6442     \PackageError{glossaries}{Option clash: 'smallcaps' and 'dua'
6443       can't both be set}{}%
6444   \else
6445     \ifglsacrsmaller
6446       \PackageError{glossaries}{Option clash: 'smaller' and 'dua'
6447         can't both be set}{}%
6448     \fi
6449   \fi
6450 \renewcommand{\newacronym}[4][[]]{%
6451   \ifx\@glsacronymlists\@empty
6452     \def\@glo@type{\acronymtype}%
6453     \setkeys{glossentry}{##1}%
6454     \DeclareAcronymList{\@glo@type}%
6455     \SetDescriptionDUAAcronymDisplayStyle{\@glo@type}%
6456   \fi
6457   \glskeylisttok{##1}%
6458   \glslabeltok{##2}%
6459   \glsshorttok{##3}%
6460   \glslongtok{##4}%
6461   \newacronymhook
6462   \DescriptionDUANewAcronymDef
6463 }%

```

Set display.

```

6464 \@for\@gls@type:=\@glsacronymlists\do{%
6465   \SetDescriptionDUAAcronymDisplayStyle{\@gls@type}%
6466 }%
6467 }%

```

AcronymDisplayStyle Sets the acronym display style for given glossary using the description setting (but not footnote or dua).

```

6468 \newcommand*\SetDescriptionAcronymDisplayStyle[1]{%
6469   \def\glsentryfmt[1]{%
6470     \ifdefempty\glscustomtext
6471     {%
6472       \ifglsused{\glslabel}%
6473       {%

```

Move the inserted text outside of \acronymfont

```

6474 \let\gls@org@insert\glsinsert
6475 \let\glsinsert\@empty
6476 \acronymfont{\gls@entryfmt}\gls@org@insert
6477 }%
6478 {%
6479 \gls@entryfmt
6480 \ifgls@has@symbol{\gls@label}%
6481 {%
6482 \gls@ifplural
6483 {%
6484 \def\@glo@symbol{\gls@entrysymbolplural{\gls@label}}%
6485 }%
6486 {%
6487 \def\@glo@symbol{\gls@entrysymbol{\gls@label}}%
6488 }%
6489 \space(\protect\firstacronymfont
6490 {\gls@caps@case
6491 {\@glo@symbol}
6492 {\@glo@symbol}
6493 {\mfirstucMakeUppercase{\@glo@symbol}}})%
6494 }%
6495 }%
6496 }%
6497 }%
6498 {\gls@customtext\glsinsert}%
6499 }%
6500 }

```

ptionNewAcronymDef

```

6501 \newcommand*{\DescriptionNewAcronymDef}{%
6502 \edef\@do@newglossaryentry{%
6503 \noexpand\newglossaryentry{\the\gls@labeltok}%
6504 {%
6505 type=\acronymtype,%
6506 name={\noexpand
6507 \acrnameformat{\the\gls@shorttok}{\the\gls@longtok}},%
6508 sort={\the\gls@shorttok},%
6509 first={\the\gls@longtok},%
6510 firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6511 text={\the\gls@shorttok},%
6512 plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6513 short={\the\gls@shorttok},%
6514 shortplural={\the\gls@shorttok\noexpand\acrpluralsuffix},%
6515 long={\the\gls@longtok},%
6516 longplural={\the\gls@longtok\noexpand\acrpluralsuffix},%
6517 symbol={\noexpand\@glo@text},%
6518 symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6519 \the\gls@keylisttok}%

```

```

6520 }%
6521 \let\@org@gl@s@assign@firstpl\gl@s@assign@firstpl
6522 \let\@org@gl@s@assign@plural\gl@s@assign@plural
6523 \let\@org@gl@s@assign@symbolplural\gl@s@assign@symbolplural
6524 \def\gl@s@assign@firstpl##1##2{%
6525   \@@gl@s@expand@field{##1}{firstpl}{##2}%
6526 }%
6527 \def\gl@s@assign@plural##1##2{%
6528   \@@gl@s@expand@field{##1}{plural}{##2}%
6529 }%
6530 \def\gl@s@assign@symbolplural##1##2{%
6531   \@@gl@s@expand@field{##1}{symbolplural}{##2}%
6532 }%
6533 \do@newglossaryentry
6534 \let\gl@s@assign@firstpl\@org@gl@s@assign@firstpl
6535 \let\gl@s@assign@plural\@org@gl@s@assign@plural
6536 \let\gl@s@assign@symbolplural\@org@gl@s@assign@symbolplural
6537 }

```

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.

```

6538 \newcommand*{\SetDescriptionAcronymStyle}{%
6539   \renewcommand{\newacronym}[4][\]{%
6540     \ifx\@gl@sacronymlists\@empty
6541       \def\@glo@type{\acronymtype}%
6542       \setkeys{glossentry}{##1}%
6543       \DeclareAcronymList{\@glo@type}%
6544       \SetDescriptionAcronymDisplayStyle{\@glo@type}%
6545     \fi
6546     \gl@keylisttok{##1}%
6547     \gl@labeltok{##2}%
6548     \gl@shorttok{##3}%
6549     \gl@longtok{##4}%
6550     \newacronymhook
6551     \DescriptionNewAcronymDef
6552   }%

```

Set display.

```

6553 \@for\@gl@s@type:=\@gl@sacronymlists\do{%
6554   \SetDescriptionAcronymDisplayStyle{\@gl@s@type}%
6555 }%

```

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.

```

6556 \ifgl@smallcaps
6557   \renewcommand{\acronymfont}[1]{\textsc{##1}}

```

```

6558 \renewcommand*{\acrpluralsuffix}{%
6559 \glstextup{\glspluralsuffix}}%
6560 \else
6561 \ifglsacrsmaller
6562 \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%
6563 \fi
6564 \fi
6565 }%

```

AcronymDisplayStyle Sets the acronym display style for given glossary with footnote setting (but not description or dua).

```

6566 \newcommand*{\SetFootnoteAcronymDisplayStyle}[1]{%
6567 \defglsentryfmt[#1]{%

```

```

6568 \ifdefempty\glscustomtext
6569 {%

```

Move the inserted text outside of \acronymfont

```

6570 \let\gls@org@insert\glsinsert
6571 \let\glsinsert\@empty
6572 \ifglsused{\glslabel}%
6573 {%
6574 \acronymfont{\glsentryfmt}\gls@org@insert
6575 }%
6576 {%
6577 \firstacronymfont{\glsentryfmt}\gls@org@insert
6578 \ifglsashaslong{\glslabel}%
6579 {%
6580 \expandafter\protect\expandafter\acrfootnote\expandafter
6581 {\@gls@link@opts}\@gls@link@label}%
6582 {%
6583 \glsifplural
6584 {\glsentrylongpl{\glslabel}}%
6585 {\glsentrylong{\glslabel}}%
6586 }%
6587 }%
6588 {}%
6589 }%
6590 }%
6591 {\glscustomtext\glsinsert}%
6592 }%
6593 }

```

FootnoteNewAcronymDef

```

6594 \newcommand*{\FootnoteNewAcronymDef}{%
6595 \edef\@do@newglossaryentry{%
6596 \noexpand\newglossaryentry{\the\glslabeltok}%
6597 {%
6598 type=\acronymtype,%

```

```

6599     name={\noexpand\acronymfont{\the\glsshorttok}},%
6600     sort={\the\glsshorttok},%
6601     text={\the\glsshorttok},%
6602     plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6603     first={\the\glsshorttok},%
6604     firstplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6605     short={\the\glsshorttok},%
6606     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6607     long={\the\glslongtok},%
6608     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6609     description={\the\glslongtok},%
6610     descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6611     \the\glskeylisttok
6612 }%
6613 }%
6614 \let\@org@gls@assign@plural\gls@assign@plural
6615 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6616 \let\@org@gls@assign@descplural\gls@assign@descplural
6617 \def\gls@assign@firstpl##1##2{%
6618   \@@gls@expand@field{##1}{firstpl}{##2}%
6619 }%
6620 \def\gls@assign@plural##1##2{%
6621   \@@gls@expand@field{##1}{plural}{##2}%
6622 }%
6623 \def\gls@assign@descplural##1##2{%
6624   \@@gls@expand@field{##1}{descplural}{##2}%
6625 }%
6626 \do@newglossaryentry
6627 \let\gls@assign@plural\@org@gls@assign@plural
6628 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6629 \let\gls@assign@descplural\@org@gls@assign@descplural
6630 }

```

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.

```

6631 \newcommand*\SetFootnoteAcronymStyle{%
6632   \renewcommand{\newacronym}[4][\]{%
6633     \ifx\@glsacronymlists\@empty
6634       \def\@glo@type{\acronymtype}%
6635       \setkeys{glossentry}{##1}%
6636       \DeclareAcronymList{\@glo@type}%
6637       \SetFootnoteAcronymDisplayStyle{\@glo@type}%
6638     \fi
6639     \glskeylisttok{##1}%
6640     \glslabeltok{##2}%
6641     \glsshorttok{##3}%
6642     \glslongtok{##4}%
6643     \newacronymhook

```

```

6644 \FootnoteNewAcronymDef
6645 }%

```

Set display

```

6646 \@for\@gls@type:=\@glsacronymlists\do{%
6647 \SetFootnoteAcronymDisplayStyle{\@gls@type}%
6648 }%

```

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.

```

6649 \ifglsacrsmallcaps
6650 \renewcommand*\acronymfont}[1]{\textsc{##1}}%
6651 \renewcommand*\acrpluralsuffix}{%
6652 \glstextup{\glspluralsuffix}}%
6653 \else
6654 \ifglsacrsmaller
6655 \renewcommand*\acronymfont}[1]{\textsmaller{##1}}%
6656 \fi
6657 \fi

```

Check for option clash

```

6658 \ifglsacrdua
6659 \PackageError{glossaries}{Option clash: ‘footnote’ and ‘dua’
6660 can’t both be set}{}%
6661 \fi
6662 }%

```

`\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.

```

6663 \DeclareRobustCommand*\glsdoparenifnotempty}[2]{%
6664 \protected@edef\gls@tmp{#1}%
6665 \ifdefempty\gls@tmp
6666 {}}%
6667 {%
6668 \ifx\gls@tmp\@gls@default@value
6669 \else
6670 \space (#2{#1})%
6671 \fi
6672 }%
6673 }

```

`\AcronymDisplayStyle` Sets the acronym display style for given glossary where neither footnote nor description is required, but smallcaps or smaller specified.

```

6674 \newcommand*\SetSmallAcronymDisplayStyle}[1]{%
6675 \defglsentryfmt[#1]{%
6676 \ifdefempty\glscustomtext
6677 {%

```

Move the inserted text outside of \acronymfont

```

6678 \let\gls@org@insert\glsinsert
6679 \let\glsinsert\@empty
6680 \ifglsused{\glslabel}%
6681 {%
6682 \acronymfont{\glsgetentryfmt}\gls@org@insert
6683 }%
6684 {%
6685 \glsgetentryfmt
6686 \ifglshassymbol{\glslabel}%
6687 {%
6688 \glsifplural
6689 {%
6690 \def\@glo@symbol{\glsentrysymbolplural{\glslabel}}%
6691 }%
6692 {%
6693 \def\@glo@symbol{\glsentrysymbol{\glslabel}}%
6694 }%
6695 \space
6696 (\glscapscase
6697 {\firstacronymfont{\@glo@symbol}}%
6698 {\firstacronymfont{\@glo@symbol}}%
6699 {\firstacronymfont{\mfirstucMakeUppercase{\@glo@symbol}}})%
6700 }%
6701 {}%
6702 }%
6703 }%
6704 {\glscustomtext\glsinsert}%
6705 }%
6706 }

```

\SmallNewAcronymDef

```

6707 \newcommand*{\SmallNewAcronymDef}{%
6708 \edef\@do@newglossaryentry{%
6709 \noexpand\newglossaryentry{\the\glslabeltok}%
6710 {%
6711 type=\acronymtype,%
6712 name={\noexpand\acronymfont{\the\glsshorttok}},%
6713 sort={\the\glsshorttok},%
6714 text={\the\glsshorttok},%

```

Default to the short plural.

```

6715 plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6716 first={\the\glslongtok},%

```

Default to the long plural.

```

6717 firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6718 short={\the\glsshorttok},%
6719 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6720 long={\the\glslongtok},%

```

```

6721     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6722     description={\noexpand\@glo@first},%
6723     descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6724     symbol={\the\glsshorttok},%

```

Default to the short plural.

```

6725     symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6726     \the\glskeylisttok
6727 }%
6728 }%
6729 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6730 \let\@org@gls@assign@plural\gls@assign@plural
6731 \let\@org@gls@assign@descplural\gls@assign@descplural
6732 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6733 \def\gls@assign@firstpl##1##2{%
6734     \@@gls@expand@field{##1}{firstpl}{##2}%
6735 }%
6736 \def\gls@assign@plural##1##2{%
6737     \@@gls@expand@field{##1}{plural}{##2}%
6738 }%
6739 \def\gls@assign@descplural##1##2{%
6740     \@@gls@expand@field{##1}{descplural}{##2}%
6741 }%
6742 \def\gls@assign@symbolplural##1##2{%
6743     \@@gls@expand@field{##1}{symbolplural}{##2}%
6744 }%
6745 \do@newglossaryentry
6746 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6747 \let\gls@assign@plural\@org@gls@assign@plural
6748 \let\gls@assign@descplural\@org@gls@assign@descplural
6749 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6750 }

```

etSmallAcronymStyle 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.

```

6751 \newcommand*\SetSmallAcronymStyle{%
6752     \renewcommand{\newacronym}[4][\]{%
6753         \ifx\@glsacronymlists\@empty
6754             \def\@glo@type{\acronymtype}%
6755             \setkeys{glossentry}{##1}%
6756             \DeclareAcronymList{\@glo@type}%
6757             \SetSmallAcronymDisplayStyle{\@glo@type}%
6758         \fi
6759         \glskeylisttok{##1}%
6760         \glslabeltok{##2}%
6761         \glsshorttok{##3}%
6762         \glslongtok{##4}%
6763         \newacronymhook
6764         \SmallNewAcronymDef
6765     }%

```


Change the display since first only contains long form.

```
6766 \@for\@gls@type:=\@glsacronymlists\do{%
6767     \SetSmallAcronymDisplayStyle{\@gls@type}%
6768 }%
```

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.

```
6769 \ifglsacrsmallcaps
6770     \renewcommand*\acronymfont}[1]{\textsc{##1}}
6771     \renewcommand*\acrpluralsuffix{%
6772         \glstextup{\glspluralsuffix}}%
6773 \else
6774     \renewcommand*\acronymfont}[1]{\textsmaller{##1}}
6775 \fi
```

check for option clash

```
6776 \ifglsacrdua
6777     \ifglsacrsmallcaps
6778         \PackageError{glossaries}{Option clash: ‘smallcaps’ and ‘dua’
6779             can’t both be set}{}%
6780     \else
6781         \PackageError{glossaries}{Option clash: ‘smaller’ and ‘dua’
6782             can’t both be set}{}%
6783     \fi
6784 \fi
6785 }%
```

\SetDUADisplayStyle Sets the acronym display style for given glossary with dua setting.

```
6786 \newcommand*\SetDUADisplayStyle}[1]{%
6787     \defglsentryfmt[#1]{\glsgenentryfmt}%
6788 }
```

\DUANewAcronymDef

```
6789 \newcommand*\DUANewAcronymDef{%
6790     \edef\@do@newglossaryentry{%
6791         \noexpand\newglossaryentry{\the\glslabeltok}%
6792         {%
6793             type=\acronymtype,%
6794             name={\the\glsshorttok},%
6795             text={\the\glslongtok},%
6796             first={\the\glslongtok},%
6797             plural={\noexpand\expandonce\noexpand\@glo@longpl},%
6798             firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6799             short={\the\glsshorttok},%
6800             shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6801             long={\the\glslongtok},%
6802             longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6803             description={\the\glslongtok},%
```

```

6804     descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6805     symbol={\the\glsshorttok},%
6806     symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6807     \the\glskeylisttok
6808 }%
6809 }%
6810 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6811 \let\@org@gls@assign@plural\gls@assign@plural
6812 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6813 \let\@org@gls@assign@descplural\gls@assign@descplural
6814 \def\gls@assign@firstpl##1##2{%
6815     \@@gls@expand@field{##1}{firstpl}{##2}%
6816 }%
6817 \def\gls@assign@plural##1##2{%
6818     \@@gls@expand@field{##1}{plural}{##2}%
6819 }%
6820 \def\gls@assign@symbolplural##1##2{%
6821     \@@gls@expand@field{##1}{symbolplural}{##2}%
6822 }%
6823 \def\gls@assign@descplural##1##2{%
6824     \@@gls@expand@field{##1}{descplural}{##2}%
6825 }%
6826 \do@newglossaryentry
6827 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6828 \let\gls@assign@plural\@org@gls@assign@plural
6829 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6830 \let\gls@assign@descplural\@org@gls@assign@descplural
6831 }

```

\SetDUASyle Always expand acronyms.

```

6832 \newcommand*{\SetDUASyle}{%
6833     \renewcommand{\newacronym}[4][]{%
6834         \ifx\@glsacronymlists\@empty
6835             \def\@glo@type{\acronymtype}%
6836             \setkeys{glossentry}{##1}%
6837             \DeclareAcronymList{\@glo@type}%
6838             \SetDUADisplayStyle{\@glo@type}%
6839         \fi
6840         \glskeylisttok{##1}%
6841         \glslabeltok{##2}%
6842         \glsshorttok{##3}%
6843         \glslongtok{##4}%
6844         \newacronymhook
6845         \DUANewAcronymDef
6846     }%
6847     \@for\@gls@type:=\@glsacronymlists\do{%
6848         \SetDUADisplayStyle{\@gls@type}%
6849     }%

```

```
6850 }
```

`\SetAcronymStyle`

```
6851 \newcommand*{\SetAcronymStyle}{%
6852   \SetDefaultAcronymStyle
6853   \ifglacrdescription
6854     \ifglacrfootnote
6855       \SetDescriptionFootnoteAcronymStyle
6856     \else
6857       \ifglacrdua
6858         \SetDescriptionDUAAcronymStyle
6859       \else
6860         \SetDescriptionAcronymStyle
6861       \fi
6862     \fi
6863   \else
6864     \ifglacrfootnote
6865       \SetFootnoteAcronymStyle
6866     \else
6867       \ifthenelse{\boolean{glacrsmallcaps}}{OR
6868         \boolean{glacrsmaller}}{%
6869         {%
6870           \SetSmallAcronymStyle
6871         }%
6872         {%
6873           \ifglacrdua
6874             \SetDUASStyle
6875           \fi
6876         }%
6877       \fi
6878     \fi
6879 }
```

Set the acronym style according to the package options

```
6880 \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.

`\SetCustomDisplayStyle` Sets the acronym display style.

```
6881 \newcommand*{\SetCustomDisplayStyle}[1]{%
6882   \defglsentryfmt[#1]{\glsgenentryfmt}%
6883 }
```

`\CustomAcronymFields`

```
6884 \newcommand*{\CustomAcronymFields}{%
```

```

6885 name={\the\glsshorttok},%
6886 description={\the\glslongtok},%
6887 first={\acrfullformat{\the\glslongtok}{\the\glsshorttok}},%
6888 firstplural={\acrfullformat
6889   {\noexpand\glsentrylongpl{\the\glslabeltok}}}%
6890   {\noexpand\glsentryshortpl{\the\glslabeltok}}},%

6891 text={\the\glsshorttok},%
6892 plural={\the\glsshorttok\noexpand\acrpluralsuffix}%
6893 }

```

CustomNewAcronymDef

```

6894 \newcommand*{\CustomNewAcronymDef}{%
6895   \protected@edef\@do@newglossaryentry{%
6896     \noexpand\newglossaryentry{\the\glslabeltok}%
6897     {%
6898       type=\acronymtype,%
6899       short={\the\glsshorttok},%
6900       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6901       long={\the\glslongtok},%
6902       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6903       user1={\the\glsshorttok},%
6904       user2={\the\glsshorttok\noexpand\acrpluralsuffix},%
6905       user3={\the\glslongtok},%
6906       user4={\the\glslongtok\noexpand\acrpluralsuffix},%
6907       \CustomAcronymFields,%
6908       \the\glskeylisttok
6909     }%
6910   }%
6911   \@do@newglossaryentry
6912 }

```

\SetCustomStyle

```

6913 \newcommand*{\SetCustomStyle}{%
6914   \renewcommand{\newacronym}[4][\]{%
6915     \ifx\@glsacronymlists\@empty
6916       \def\@glo@type{\acronymtype}%
6917       \setkeys{glossentry}{##1}%
6918       \DeclareAcronymList{\@glo@type}%
6919       \SetCustomDisplayStyle{\@glo@type}%
6920     \fi
6921     \glskeylisttok{##1}%
6922     \glslabeltok{##2}%
6923     \glsshorttok{##3}%
6924     \glslongtok{##4}%
6925     \newacronymhook
6926     \CustomNewAcronymDef
6927   }%

```

Set the display

```

6928 \@for\@gls@type:=\@glsacronymlists\do{%
6929     \SetCustomDisplayStyle{\@gls@type}%
6930 }%
6931 }

```

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.

```

6932 \RequirePackage{glossary-hypernav}

```

The styles that use list-like environments. These are not loaded if the `nolist` option is used:

```

6933 \@gls@loadlist

```

The styles that use the `longtable` environment. These are not loaded if the `no-long package` option is used.

```

6934 \@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.

```

6935 \@gls@loadsuper

```

The tree-like styles. These are not loaded if the `notree` package option is used.

```

6936 \@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.

```

6937 \ifx\@glossary@default@style\relax
6938 \else
6939   \setglossarystyle{\@glossary@default@style}
6940 \fi

```

1.19 Debugging Commands

`\showgloparent` `\showgloparent{<label>}`

```

6941 \newcommand*{\showgloparent}[1]{%
6942   \expandafter\show\csname glo@\glsdetoklabel{#1}@parent\endcsname
6943 }

```

`\showglolevel` `\showglolevel{<label>}`

```

6944 \newcommand*{\showglolevel}[1]{%
6945   \expandafter\show\csname glo@\glsdetoklabel{#1}@level\endcsname
6946 }

```

\showglolevel \showglolevel{<label>}

```

6947 \newcommand*{\showglolevel}[1]{%
6948   \expandafter\show\csname glo@\glsdetoklabel{#1}@level\endcsname
6949 }

```

\showgloplural \showgloplural{<label>}

```

6950 \newcommand*{\showgloplural}[1]{%
6951   \expandafter\show\csname glo@\glsdetoklabel{#1}@plural\endcsname
6952 }

```

\showglofirst \showglofirst{<label>}

```

6953 \newcommand*{\showglofirst}[1]{%
6954   \expandafter\show\csname glo@\glsdetoklabel{#1}@first\endcsname
6955 }

```

\showglofirstpl \showglofirstpl{<label>}

```

6956 \newcommand*{\showglofirstpl}[1]{%
6957   \expandafter\show\csname glo@\glsdetoklabel{#1}@firstpl\endcsname
6958 }

```

\showgloftype \showgloftype{<label>}

```

6959 \newcommand*{\showgloftype}[1]{%
6960   \expandafter\show\csname glo@\glsdetoklabel{#1}@type\endcsname
6961 }

```

\showglocounter \showglocounter{<label>}

```

6962 \newcommand*{\showglocounter}[1]{%
6963   \expandafter\show\csname glo@\glsdetoklabel{#1}@counter\endcsname
6964 }

```

\showglouserii \showglouserii{<label>}

```

6965 \newcommand*{\showglouserii}[1]{%
6966   \expandafter\show\csname glo@\glsdetoklabel{#1}@userii\endcsname
6967 }

```

\showglouseriii \showglouseriii{<label>}

```

6968 \newcommand*{\showglouseriii}[1]{%
6969   \expandafter\show\csname glo@\glsdetoklabel{#1}@useriii\endcsname
6970 }

```

\showglouseriv \showglouseriv{<label>}

```

6971 \newcommand*{\showglouseriv}[1]{%
6972   \expandafter\show\csname glo@\glsdetoklabel{#1}@useriv\endcsname
6973 }

```

\showglouserv \showglouserv{<label>}

```

6974 \newcommand*{\showglouserv}[1]{%
6975   \expandafter\show\csname glo@\glsdetoklabel{#1}@userv\endcsname
6976 }

```

\showglouservi \showglouservi{<label>}

```

6977 \newcommand*{\showglouservi}[1]{%
6978   \expandafter\show\csname glo@\glsdetoklabel{#1}@uservi\endcsname
6979 }

```

```

6980 \newcommand*{\showglouservi}[1]{%
6981   \expandafter\show\csname glo@\glsdetoklabel{#1}@uservi\endcsname
6982 }

```

\showglongame \showglongame{\label}

```

6983 \newcommand*{\showglongame}[1]{%
6984   \expandafter\show\csname glo@\glsdetoklabel{#1}@name\endcsname
6985 }

```

\showglodesc \showglodesc{\label}

```

6986 \newcommand*{\showglodesc}[1]{%
6987   \expandafter\show\csname glo@\glsdetoklabel{#1}@desc\endcsname
6988 }

```

\showglodescplural \showglodescplural{\label}

```

6989 \newcommand*{\showglodescplural}[1]{%
6990   \expandafter\show\csname glo@\glsdetoklabel{#1}@descplural\endcsname
6991 }

```

\showglosort \showglosort{\label}

```

6992 \newcommand*{\showglosort}[1]{%
6993   \expandafter\show\csname glo@\glsdetoklabel{#1}@sort\endcsname
6994 }

```

\showglosymbol \showglosymbol{\label}

```

6995 \newcommand*{\showglosymbol}[1]{%
6996   \expandafter\show\csname glo@\glsdetoklabel{#1}@symbol\endcsname
6997 }

```

\showglosymbolplural \showglosymbolplural{\label}


```

6998 \newcommand*{\showglosymbolplural}[1]{%
6999   \expandafter\show\csname glo@\glsdetoklabel{#1}@symbolplural\endcsname
7000 }

```

\showgloshort \showgloshort{<label>}

```

7001 \newcommand*{\showgloshort}[1]{%
7002   \expandafter\show\csname glo@\glsdetoklabel{#1}@short\endcsname
7003 }

```

\showglolong \showglolong{<label>}

```

7004 \newcommand*{\showglolong}[1]{%
7005   \expandafter\show\csname glo@\glsdetoklabel{#1}@long\endcsname
7006 }

```

\showgloindex \showgloindex{<label>}

```

7007 \newcommand*{\showgloindex}[1]{%
7008   \expandafter\show\csname glo@\glsdetoklabel{#1}@index\endcsname
7009 }

```

\showgloflag \showgloflag{<label>}

```

7010 \newcommand*{\showgloflag}[1]{%
7011   \expandafter\show\csname ifglo@\glsdetoklabel{#1}@flag\endcsname
7012 }

```

\showgloloclist \showgloloclist{<label>}

```

7013 \newcommand*{\showgloloclist}[1]{%
7014   \expandafter\show\csname glo@\glsdetoklabel{#1}@loclist\endcsname
7015 }

```

\showacronymlists \showacronymlists

Show list of glossaries that have been flagged as a list of acronyms.

```

7016 \newcommand*{\showacronymlists}{%
7017   \show\@glsacronymlists
7018 }

```

\showglossaries \showglossaries

Show list of defined glossaries.

```

7019 \newcommand*{\showglossaries}{%
7020   \show\@glo@types
7021 }

```

\showglossaryin \showglossaryin{<glossary-label>}

Show the ‘in’ extension for the given glossary.

```

7022 \newcommand*{\showglossaryin}[1]{%
7023   \expandafter\show\csname @glo@#1@in\endcsname
7024 }

```

\showglossaryout \showglossaryout{<glossary-label>}

Show the ‘out’ extension for the given glossary.

```

7025 \newcommand*{\showglossaryout}[1]{%
7026   \expandafter\show\csname @glo@#1@out\endcsname
7027 }

```

\showglossarytitle \showglossarytitle{<glossary-label>}

Show the title for the given glossary.

```

7028 \newcommand*{\showglossarytitle}[1]{%
7029   \expandafter\show\csname @glo@#1@title\endcsname
7030 }

```

\showglossarycounter \showglossarycounter{<glossary-label>}

Show the counter for the given glossary.

```

7031 \newcommand*{\showglossarycounter}[1]{%
7032   \expandafter\show\csname @glo@#1@counter\endcsname
7033 }

```

\showglossaryentries \showglossaryentries{<glossary-label>}

Show the list of entry labels for the given glossary.

```
7034 \newcommand*{\showglossaryentries}[1]{%
7035   \expandafter\show\csname glo@list@#1\endcsname
7036 }
```

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.

```
7037 \csname ifglscpatible-2.07\endcsname
7038   \RequirePackage{glossaries-compatible-207}
7039 \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.

```
7040 \NeedsTeXFormat{LaTeX2e}
7041 \ProvidesPackage{glossaries-prefix}[2014/07/30 v4.08 (NLCT)]
```

Pass all options to glossaries:

```
7042 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{glossaries}}
```

Process options:

```
7043 \ProcessOptions
```

Load glossaries:

```
7044 \RequirePackage{glossaries}
```

Add the new keys:

```
7045 \define@key{glossentry}{prefixfirst}{\def\@glo@entryprefixfirst{#1}}%
7046 \define@key{glossentry}{prefixfirstplural}{\def\@glo@entryprefixfirstplural{#1}}%
7047 \define@key{glossentry}{prefix}{\def\@glo@entryprefix{#1}}%
7048 \define@key{glossentry}{prefixplural}{\def\@glo@entryprefixplural{#1}}%
```

Add them to \@gls@keymap:

```
7049 \appto\@gls@keymap{,%  
7050   {prefixfirst}{prefixfirst},%  
7051   {prefixfirstplural}{prefixfirstplural},%  
7052   {prefix}{prefix},%  
7053   {prefixplural}{prefixplural}}%  
7054 }
```

Set the default values:

```
7055 \appto\@newglossaryentryprehook{%  
7056   \def\@glo@entryprefix{}%  
7057   \def\@glo@entryprefixplural{}%  
7058   \let\@glo@entryprefixfirst\@gls@default@value  
7059   \let\@glo@entryprefixfirstplural\@gls@default@value  
7060 }
```

Set the assignment code:

```
7061 \appto\@newglossaryentryposthook{%  
7062   \gls@assign@field{}\@glo@label{prefix}\@glo@entryprefix}%  
7063   \gls@assign@field{}\@glo@label{prefixplural}\@glo@entryprefixplural}%  
7064 }
```

If prefixfirst has not been supplied, make it the same as prefix.

```
7064 \expandafter\gls@assign@field\expandafter  
7065   {\csname glo@\@glo@label @prefix\endcsname}\@glo@label{prefixfirst}%  
7066   {\@glo@entryprefixfirst}}%
```

If prefixfirstplural has not been supplied, make it the same as prefixplural.

```
7067 \expandafter\gls@assign@field\expandafter  
7068   {\csname glo@\@glo@label @prefixplural\endcsname}\@glo@label}%  
7069   {prefixfirstplural}\@glo@entryprefixfirstplural}}%  
7070 }
```

Define commands to access these fields:

glsentryprefixfirst

```
7071 \newcommand*\glsentryprefixfirst[1]{\csuse{glo@#1@prefixfirst}}
```

glsentryprefixfirstplural

```
7072 \newcommand*\glsentryprefixfirstplural[1]{\csuse{glo@#1@prefixfirstplural}}
```

\glsentryprefix

```
7073 \newcommand*\glsentryprefix[1]{\csuse{glo@#1@prefix}}
```

\glsentryprefixplural

```
7074 \newcommand*\glsentryprefixplural[1]{\csuse{glo@#1@prefixplural}}
```

Now for the initial upper case variants:

Glsentryprefixfirst

```
7075 \newrobustcmd*\Glsentryprefixfirst[1]{%  
7076   \protected@edef\@glo@text{\csname glo@#1@prefixfirst\endcsname}%  
7077   \xmakefirstuc\@glo@text  
7078 }
```

ryprefixfirstplural

```
7079 \newrobustcmd*{\Glsentryprefixfirstplural}[1]{%
7080   \protected@edef\@glo@text{\csname glo@#1@prefixfirstplural\endcsname}%
7081   \xmakefirstuc\@glo@text
7082 }
```

\Glsentryprefix

```
7083 \newrobustcmd*{\Glsentryprefix}[1]{%
7084   \protected@edef\@glo@text{\csname glo@#1@prefix\endcsname}%
7085   \xmakefirstuc\@glo@text
7086 }
```

lsentryprefixplural

```
7087 \newrobustcmd*{\Glsentryprefixplural}[1]{%
7088   \protected@edef\@glo@text{\csname glo@#1@prefixplural\endcsname}%
7089   \xmakefirstuc\@glo@text
7090 }
```

Define commands to determine if the prefix keys have been set:

\ifglshasprefix

```
7091 \newcommand*{\ifglshasprefix}[3]{%
7092   \ifcempty{glo@#1@prefix}%
7093   {#3}%
7094   {#2}%
7095 }
```

ifglshasprefixplural

```
7096 \newcommand*{\ifglshasprefixplural}[3]{%
7097   \ifcempty{glo@#1@prefixplural}%
7098   {#3}%
7099   {#2}%
7100 }
```

ifglshasprefixfirst

```
7101 \newcommand*{\ifglshasprefixfirst}[3]{%
7102   \ifcempty{glo@#1@prefixfirst}%
7103   {#3}%
7104   {#2}%
7105 }
```

asprefixfirstplural

```
7106 \newcommand*{\ifglshasprefixfirstplural}[3]{%
7107   \ifcempty{glo@#1@prefixfirstplural}%
7108   {#3}%
7109   {#2}%
7110 }
```

Define commands that insert the prefix before commands like \gls:

```
\pgls
7111 \newrobustcmd{\pgls}{\@gls@hyp@opt\@pgls}
```

\@pgls Unstarred version.

```
7112 \newcommand*{\@pgls}[2][\%
7113 \new@ifnextchar[%
7114 {\@pgls@{#1}{#2}}%
7115 {\@pgls@{#1}{#2}[]}%
7116 }
```

\@pgls@ Read in the final optional argument:

```
7117 \def\@pgls@#1#2[#3]{%
7118 \glsdoifexists{#2}%
7119 {%
7120 \ifglsused{#2}%
7121 {%
7122 \glsentryprefix{#2}%
7123 }%
7124 {%
7125 \glsentryprefixfirst{#2}%
7126 }%
7127 \@gls@{#1}{#2}[#3]%
7128 }%
7129 }
```

Similarly for the plural version:

```
\pglsp1
7130 \newrobustcmd{\pglsp1}{\@gls@hyp@opt\@pglsp1}
```

\@pglsp1 Unstarred version.

```
7131 \newcommand*{\@pglsp1}[2][\%
7132 \new@ifnextchar[%
7133 {\@pglsp1@{#1}{#2}}%
7134 {\@pglsp1@{#1}{#2}[]}%
7135 }
```

\@pglsp1@ Read in the final optional argument:

```
7136 \def\@pglsp1@#1#2[#3]{%
7137 \glsdoifexists{#2}%
7138 {%
7139 \ifglsused{#2}%
7140 {%
7141 \glsentryprefixplural{#2}%
7142 }%
7143 {%
7144 \glsentryprefixfirstplural{#2}%
7145 }
```

```

7145 }%
7146 \@glspl@{#1}{#2}[#3]%
7147 }%
7148 }

```

Now for the first letter upper case versions:

`\Pgl`

```

7149 \newrobustcmd{\Pgl}{\@gls@hyp@opt\Pgl}

```

`\@Pgl` Unstarred version.

```

7150 \newcommand*{\@Pgl}[2][]{%
7151   \new@ifnextchar[%
7152     {\@Pgl@{#1}{#2}}%
7153     {\@Pgl@{#1}{#2}[]}%
7154 }

```

`\@Pgl@` Read in the final optional argument:

```

7155 \def\@Pgl@#1#2[#3]{%
7156   \glsdoifexists{#2}%
7157   {%
7158     \ifglsused{#2}%
7159     {%
7160       \ifglshasprefix{#2}%
7161       {%
7162         \Glsentryprefix{#2}%
7163         \@gls@{#1}{#2}[#3]%
7164       }%
7165       {\@Gls@{#1}{#2}[#3]}%
7166     }%
7167     {%
7168       \ifglshasprefixfirst{#2}%
7169       {%
7170         \Glsentryprefixfirst{#2}%
7171         \@gls@{#1}{#2}[#3]%
7172       }%
7173       {\@Gls@{#1}{#2}[#3]}%
7174     }%
7175   }%
7176 }

```

Similarly for the plural version:

`\Pglpl`

```

7177 \newrobustcmd{\Pglpl}{\@gls@hyp@opt\Pglpl}

```

`\@Pglpl` Unstarred version.

```

7178 \newcommand*{\@Pglpl}[2][]{%

```

```

7179 \new@ifnextchar[%
7180 {\@Pglsp1@{#1}{#2}}%
7181 {\@Pglsp1@{#1}{#2}[]}%
7182 }

```

\@Pglsp1@ Read in the final optional argument:

```

7183 \def\@Pglsp1@#1#2[#3]{%
7184 \glsdoifexists{#2}%
7185 {%
7186 \ifglused{#2}%
7187 {%
7188 \ifglshasprefixplural{#2}%
7189 {%
7190 \Glsentryprefixplural{#2}%
7191 \@glsp1@{#1}{#2}[#3]%
7192 }%
7193 {\@Glspl@{#1}{#2}[#3]}%
7194 }%
7195 {%
7196 \ifglshasprefixfirstplural{#2}%
7197 {%
7198 \Glsentryprefixfirstplural{#2}%
7199 \@glsp1@{#1}{#2}[#3]%
7200 }%
7201 {\@Glspl@{#1}{#2}[#3]}%
7202 }%
7203 }%
7204 }

```

Finally the all upper case versions:

\PGLS

```

7205 \newrobustcmd{\PGLS}{\@gls@hyp@opt\@PGLS}

```

\@PGLS Unstarred version.

```

7206 \newcommand*{\@PGLS}[2][]{%
7207 \new@ifnextchar[%
7208 {\@PGLS@{#1}{#2}}%
7209 {\@PGLS@{#1}{#2}[]}%
7210 }

```

\@PGLS@ Read in the final optional argument:

```

7211 \def\@PGLS@#1#2[#3]{%
7212 \glsdoifexists{#2}%
7213 {%
7214 \ifglused{#2}%
7215 {%
7216 \mfirstucMakeUppercase{\glentryprefix{#2}}%

```



```

7217 }%
7218 {%
7219 \mfirstucMakeUppercase{\glstryentryprefixfirst{#2}}}%
7220 }%
7221 \@GLS@{#1}{#2}[#3]%
7222 }%
7223 }

```

Plural version:

\PGLSp1

```

7224 \newrobustcmd{\PGLSp1}{\@glshyp@opt\PGLSp1}

```

\@PGLSp1 Unstarred version.

```

7225 \newcommand*{\@PGLSp1}[2][{}]{%
7226 \new@ifnextchar[%
7227 {\@PGLSp1@{#1}{#2}}}%
7228 {\@PGLSp1@{#1}{#2}[]}%
7229 }

```

\@PGLSp1@ Read in the final optional argument:

```

7230 \def\@PGLSp1@#1#2[#3]{%
7231 \glstryentryexists{#2}%
7232 {%
7233 \ifglstryused{#2}%
7234 {%
7235 \mfirstucMakeUppercase{\glstryentryprefixplural{#2}}}%
7236 }%
7237 {%
7238 \mfirstucMakeUppercase{\glstryentryprefixfirstplural{#2}}}%
7239 }%
7240 \@GLS@{#1}{#2}[#3]%
7241 }%
7242 }

```

3 Mfirstuc Documented Code

```

7243 \NeedsTeXFormat{LaTeX2e}
7244 \ProvidesPackage{mfirstuc}[2014/07/30 v1.09 (NLCT)]

```

Requires etoolbox:

```

7245 \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`.

```

7246 \newif\if@glscs
7247 \newtoks\@glsmfirst
7248 \newtoks\@glsmrest
7249 \newrobustcmd*{\makefirstuc}[1]{%
7250   \def\gls@argi{#1}%
7251   \ifx\gls@argi\@empty

    If the argument is empty, do nothing.

7252   \else

7253     \def\@gls@tmp{\ #1}%
7254     \@onelevel@sanitize\@gls@tmp
7255     \expandafter\@gls@checkcs\@gls@tmp\relax\relax
7256     \if@glscs
7257       \@gls@getbody #1{}\@nil
7258       \ifx\@gls@rest\@empty
7259         \glsmakefirstuc{#1}%
7260       \else
7261         \expandafter\@gls@split\@gls@rest\@nil
7262         \ifx\@gls@first\@empty
7263           \glsmakefirstuc{#1}%
7264         \else
7265           \expandafter\@glsmfirst\expandafter{\@gls@first}%
7266           \expandafter\@glsmrest\expandafter{\@gls@rest}%
7267           \edef\@gls@domfirstuc{\noexpand\@gls@body
7268             {\noexpand\glsmakefirstuc\the\@glsmfirst}%
7269             \the\@glsmrest}%
7270           \@gls@domfirstuc
7271         \fi
7272       \fi
7273     \else
7274       \glsmakefirstuc{#1}%
7275     \fi
7276   \fi
7277 }
```

Put first argument in `\@gls@first` and second argument in `\@gls@rest`:

```

7278 \def\@gls@split#1#2\@nil{%
7279   \def\@gls@first{#1}\def\@gls@rest{#2}%
7280 }

7281 \def\@gls@checkcs#1 #2#3\relax{%
7282   \def\@gls@argi{#1}\def\@gls@argii{#2}%
7283   \ifx\@gls@argi\@gls@argii
7284     \@glscstrue
7285   \else
7286     \@glscsfalse
```

```

7287 \fi
7288 }

\@gls@makefirstuc  Make first thing upper case:
7289 \def\@gls@makefirstuc#1{\mfirstucMakeUppercase #1}

irstucMakeUppercase  Allow user to replace \MakeUppercase with another case changing command.
7290 \newcommand*{\mfirstucMakeUppercase}{\MakeUppercase}

\glsmakefirstuc  Provide a user command to make it easier to customise.
7291 \newcommand*{\glsmakefirstuc}[1]{\@gls@makefirstuc{#1}}

    Get the first grouped argument and store in \@gls@body.
7292 \def\@gls@getbody#1#{\def\@gls@body{#1}\@gls@gobbletonil}

    Scoup up everything to \@nil and store in \@gls@rest:
7293 \def\@gls@gobbletonil#1\@nil{\def\@gls@rest{#1}}

\xmakefirstuc  Expand argument once before applying \makefirstuc (added v1.01).
7294 \newcommand*{\xmakefirstuc}[1]{%
7295 \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).
7296 \newrobustcmd*{\capitalisewords}[1]{%
7297 \def\gls@add@space{%
7298 \let\@mfu@domakefirstuc\makefirstuc
7299 \let\@mfu@checkword\@gobble
7300 \mfu@capitalisewords#1 \@nil\mfu@endcap
7301 }

7302 \def\mfu@capitalisewords#1 #2\mfu@endcap{%
7303 \def\mfu@cap@first{#1}%
7304 \def\mfu@cap@second{#2}%
7305 \gls@add@space
7306 \@mfu@checkword{#1}%
7307 \@mfu@domakefirstuc{#1}%
7308 \def\gls@add@space{ }%
7309 \ifx\mfu@cap@second\@nnil
7310 \let\next@mfu@cap\mfu@noop
7311 \else
7312 \let\next@mfu@cap\mfu@capitalisewords
7313 \let\@mfu@checkword\mfu@checkword
7314 \fi
7315 \next@mfu@cap#2\mfu@endcap
7316 }
7317 \def\mfu@noop#1\mfu@endcap{}
```

`\mfu@checkword` Check if word should be capitalised.

```
7318 \newcommand*\mfu@checkword[1]{%
7319   \ifinlist{#1}{\@mfu@nocaplist}%
7320   {%
7321     \let\@mfu@domakefirstuc\@firstofone
7322   }%
7323   {%
7324     \let\@mfu@domakefirstuc\makefirstuc
7325   }%
7326 }
```

`\@mfu@nocaplist` List of words that shouldn't be capitalised.

```
7327 \newcommand*\@mfu@nocaplist{}
```

`\MFUnocap` Provide the user with a means to add a word to the list.

```
7328 \newcommand*\MFUnocap[1]{\listadd{\@mfu@nocaplist}{#1}}
```

`\gMFUnocap` Global version.

```
7329 \newcommand*\gMFUnocap[1]{\listgadd{\@mfu@nocaplist}{#1}}
```

`\MFUclear` Clear the list

```
7330 \newcommand*\MFUclear{\renewcommand*\@mfu@nocaplist{}}
```

`\xcapitalisewords` Short-cut command:

```
7331 \newcommand*\xcapitalisewords[1]{%
7332   \expandafter\capitalisewords\expandafter{#1}%
7333 }
```

4 Mfirstuc-english Documented Code

```
7334 \NeedsTeXFormat{LaTeX2e}
7335 \ProvidesPackage{mfirstuc-english}[2014/07/30 v1.0 (NLCT)]
```

Load mfirstuc if not already loaded:

```
7336 \RequirePackage{mfirstuc}
```

Add no-cap words. (List isn't a complete list.)

```
7337 \MFUnocap{a}
7338 \MFUnocap{an}
7339 \MFUnocap{and}
7340 \MFUnocap{but}
7341 \MFUnocap{for}
7342 \MFUnocap{in}
7343 \MFUnocap{of}
7344 \MFUnocap{or}
7345 \MFUnocap{no}
7346 \MFUnocap{nor}
7347 \MFUnocap{so}
```

```

7348 \MFUnocap{some}
7349 \MFUnocap{the}
7350 \MFUnocap{with}
7351 \MFUnocap{yet}

```

5 Glossary Styles

5.1 Glossary hyper-navigation definitions (glossary-hypernav package)

Package Definition:

```

7352 \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`

```

7353 \newcommand*{\glsnavhyperlink}[3][\@glo@type]{%
7354   \edef\gls@grplabel{#2}\protected@edef\gls@grptitle{#3}%
7355   \@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`

```

7356 \newcommand*{\glsnavhypertarget}[3][\@glo@type]{%
  Add this group to the aux file for re-run check.
7357   \protected@write\@auxout{}{\string\@gls@hypergroup{#1}{#2}}%
  Add the target.
7358   \@glstarget{glsn:#1@#2}{#3}%
  Check list of know groups to determine if a re-run is required.
7359   \expandafter\let
7360     \expandafter\@gls@list\csname @gls@hypergroup@list@#1\endcsname
  Iterate through list and terminate loop if this group is found.
7361   \@for\@gls@elem:=\@gls@list\do{%
7362     \ifthenelse{\equal{\@gls@elem}{#2}}{\@endfortrue}{}%

```

Check if list terminated prematurely.

```
7363 \if@endfor
7364 \else
```

This group was not included in the list, so issue a warning.

```
7365 \GlossariesWarningNoLine{Navigation panel
7366   for glossary type ‘#1’^^Jmissing group ‘#2’}%
7367 \gdef\gls@hypergrouprerun{%
7368   \GlossariesWarningNoLine{Navigation panel
7369   has changed. Rerun LaTeX}}%
7370 \fi
7371 }
```

`\gls@hypergrouprerun` Give a warning at the end if re-run required

```
7372 \let\gls@hypergrouprerun\relax
7373 \AtEndDocument{\gls@hypergrouprerun}
```

`\@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.

```
7374 \newcommand*{\@gls@hypergroup}[2]{%
7375 \ifundefined{\@gls@hypergrouplist@#1}{%
7376   \expandafter\xdef\csname @gls@hypergrouplist@#1\endcsname{#2}%
7377 }{%
7378   \expandafter\let\expandafter\@gls@tmp
7379   \csname @gls@hypergrouplist@#1\endcsname
7380   \expandafter\xdef\csname @gls@hypergrouplist@#1\endcsname{%
7381     \@gls@tmp,#2}%
7382 }%
7383 }
```

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`

```
7384 \newcommand*{\glsnavigation}{%
7385 \def\@gls@between{%
7386 \@ifundefined{\@gls@hypergrouplist@\@glo@type}{%
7387   \def\@gls@list{%
7388 }{%
7389   \expandafter\let\expandafter\@gls@list
7390   \csname @gls@hypergrouplist@\@glo@type\endcsname
7391 }%}
```

```

7392 \@for\@gls@tmp:=\@gls@list\do{%
7393   \@gls@between

7394   \@gls@getgrouptitle{\@gls@tmp}{\@gls@grptitle}%
7395   \glsnavhyperlink{\@gls@tmp}{\@gls@grptitle}%
7396   \let\@gls@between\glshypernavsep%
7397 }%
7398 }

```

`\glshypernavsep` Separator for the hyper navigation bar.

```

7399 \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`

```

7400 \newcommand*\glssymbolnav{%
7401 \glsnavhyperlink{glssymbols}{\glsgetgrouptitle{glssymbols}}%
7402 \glshypernavsep
7403 \glsnavhyperlink{glsnumbers}{\glsgetgrouptitle{glsnumbers}}%
7404 \glshypernavsep
7405 }

```

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.

```

7406 \ProvidesPackage{glossary-inline}[2013/11/14 v4.0 (NLCT)]

```

`inline` Define the inline style.

```

7407 \newglossarystyle{inline}{%

```

Start of glossary sets up first empty separator between entries. (This is then changed by `\glossentry`)

```

7408   \renewenvironment{theglossary}%
7409   {%
7410     \def\gls@inlinesep{}%
7411     \def\gls@inlinesubsep{}%
7412     \def\gls@inlinepostchild{}%
7413   }%
7414   {\glspostinline}%

```

No header:

```

7415 \renewcommand*\glossaryheader{}%

```

No group headings (if heading is required, add `\glsinlinedopostchild` to start definition in case heading follows a child entry):

```

7416 \renewcommand*\glsgroupheading[1]{}%

```

Just display separator followed by name and description:

```

7417 \renewcommand{\glossentry}[2]{%
7418   \glsinlinedopostchild
7419   \gls@inlinesep
7420   \glsentryitem{##1}%
7421   \glsinlinenameformat{##1}{%
7422     \glossentryname{##1}%
7423   }%
7424   \ifglshasdescsuppressed{##1}%
7425   {%
7426     \glsinlineemptydescformat
7427     {%
7428       \glossentrysymbol{##1}%
7429     }%
7430     {%
7431       ##2%
7432     }%
7433   }%
7434   {%
7435     \ifglshasdesc{##1}%
7436     {\glsinlinedescformat{\glossentrydesc{##1}}{\glossentrysymbol{##1}}{##2}}%
7437     {\glsinlineemptydescformat{\glossentrysymbol{##1}}{##2}}%
7438   }%
7439   \ifglshaschildren{##1}%
7440   {%
7441     \glsresetsubentrycounter
7442     \glsinlineparentchildseparator
7443     \def\gls@inlinesubsep{%
7444       \def\gls@inlinepostchild{\glsinlinepostchild}%
7445     }%
7446     {}%
7447   \def\gls@inlinesep{\glsinlineseparator}%
7448 }%
```

Sub-entries display description:

```

7449 \renewcommand{\subglossentry}[3]{%
7450   \gls@inlinesubsep%
7451   \glsinlinesubnameformat{##2}{%
7452     \glossentryname{##2}}%
7453   \glsentryitem{##2}%
7454   \glsinlinesubdescformat{\glossentrydesc{##2}}{\glossentrysymbol{##2}}{##3}%
7455   \def\gls@inlinesubsep{\glsinlinesubseparator}%
7456 }%
```

Nothing special between groups:

```

7457 \renewcommand*{\glsgroupskip}{}%
7458 }
```

`\glsinlinedopostchild`

```

7459 \newcommand*{\glsinlinedopostchild}{%
```



```

7460 \gls@inlinepostchild
7461 \def\gls@inlinepostchild{}%
7462 }

\glsinlineseparator Separator to use between entries.
7463 \newcommand*\glsinlineseparator{}{\space}

\glsinlinesubseparator Separator to use between sub-entries.
7464 \newcommand*\glsinlinesubseparator}{,\space}

\glsinlineparentchildseparator Separator to use between parent and children.
7465 \newcommand*\glsinlineparentchildseparator}{:\space}

\glsinlinepostchild Hook to use between child and next entry
7466 \newcommand*\glsinlinepostchild{}{}

\glsinlinepostinline Terminator for inline glossary.
7467 \newcommand*\glsinlinepostinline{}\glsinlinepostdescription\space}

\glsinlinenameformat Formats the name of the entry (first argument label, second argument name):
7468 \newcommand*\glsinlinenameformat}[2]{\glstarget{#1}{#2}}

\glsinlinedescformat Formats the entry's description, symbol and location list:
7469 \newcommand*\glsinlinedescformat}[3]{\space#1}

\glsinlineemptydescformat Formats the entry's symbol and location list when the description is empty:
7470 \newcommand*\glsinlineemptydescformat}[2]{}

\glsinlinesubnameformat Formats the name of the subentry (first argument label, second argument
name):
7471 \newcommand*\glsinlinesubnameformat}[2]{\glstarget{#1}{}}

\glsinlinesubdescformat Formats the subentry's description, symbol and location list:
7472 \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.

```
7473 \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.

```
7474 \newglossarystyle{list}{%
```

Use description environment:

```
7475 \renewenvironment{theglossary}%
7476 {\begin{description}}{\end{description}}%
```

No header at the start of the environment:

```
7477 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7478 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries start a new item in the list:

```
7479 \renewcommand*{\glossentry}[2]{%
7480 \item[\glsentryitem{##1}%
7481 \glstarget{##1}{\glossentryname{##1}}]
7482 \glossentrydesc{##1}\glspostdescription\space ##2}%
```

Sub-entries continue on the same line:

```
7483 \renewcommand*{\subglossentry}[3]{%
7484 \glssubentryitem{##2}%
7485 \glstarget{##2}{\strut}%
7486 \glossentrydesc{##2}\glspostdescription\space ##3.}%
7487 % \end{macrocode}
7488 % Add vertical space between groups:
7489 %\changes{3.03}{2012/09/21}{added check for glsnogroupskip}
7490 % \begin{macrocode}
7491 \renewcommand*{\glsgroupskip}{\ifglsgroupskip\else\indexspace\fi}%
7492 }
```

listgroup The listgroup style is like the list style, but the glossary groups have headings.

```
7493 \newglossarystyle{listgroup}{%
```

Base it on the list style:

```
7494 \setglossarystyle{list}%
```

Each group has a heading:

```
7495 \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.

```
7496 \newglossarystyle{listhypergroup}{%
```

Base it on the list style:

```
7497 \setglossarystyle{list}%
```

Add navigation links at the start of the environment:

```
7498 \renewcommand*{\glossaryheader}{%
7499 \item[\glsnavigation]}%
```

Each group has a heading with a hypertarget:

```
7500 \renewcommand*{\glsgroupheading}[1]{%
7501 \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.

```
7502 \newglossarystyle{altlist}{%
```

Base it on the list style:

```
7503 \setglossarystyle{list}%
```

Main (level 0) entries start a new item in the list with a line break after the entry name:

```
7504 \renewcommand*{\glossentry}[2]{%
7505 \item[\glsentryitem{##1}%
7506 \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.

```
7507 \mbox{}\par\nobreak\@afterheading
7508 \glossentrydesc{##1}\glspostdescription\space ##2}%
```

Sub-entries start a new paragraph:

```
7509 \renewcommand{\subglossentry}[3]{%
7510 \par
7511 \glssubentryitem{##2}%
7512 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space ##3}%
7513 }
```

altlistgroup The altlistgroup glossary style is like the altlist style, but the glossary groups have headings.

```
7514 \newglossarystyle{altlistgroup}{%
```

Base it on the altlist style:

```
7515 \setglossarystyle{altlist}%
```

Each group has a heading:

```
7516 \renewcommand*{\glsgroupheading}[1]{\item[\glsgetgrouptitle{##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.

```
7517 \newglossarystyle{altlisthypergroup}{%
```

Base it on the altlist style:

```
7518 \setglossarystyle{altlist}%
```

Add navigation links at the start of the environment:

```
7519 \renewcommand*{\glossaryheader}{%
7520 \item[\glsnavigation]}%
```

Each group has a heading with a hypertarget:

```
7521 \renewcommand*{\glsgroupheading}[1]{%
7522 \item[\glsnavhypertarget{##1}{\glsgetgrouptitle{##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.

```
7523 \newglossarystyle{listdotted}{%
```

Base it on the list style:

```
7524 \setglossarystyle{list}%
```

Each main (level 0) entry starts a new item:

```
7525 \renewcommand*{\glossentry}[2]{%
7526 \item[]\makebox[\glslistdottedwidth][l]{%
7527 \glssentryitem{##1}%
7528 \glstarget{##1}{\glossentryname{##1}}%
7529 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}\glossentrydesc{##1}}%
```

Sub entries have the same format as main entries:

```
7530 \renewcommand*{\subglossentry}[3]{%
7531 \item[]\makebox[\glslistdottedwidth][l]{%
7532 \glssubentryitem{##2}%
7533 \glstarget{##2}{\glossentryname{##2}}%
7534 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}\glossentrydesc{##2}}%
7535 }
```

`\glslistdottedwidth`

```
7536 \newlength\glslistdottedwidth
7537 \setlength{\glslistdottedwidth}{.5\hsize}
```

sublistdotted This style is similar to the `glostylelistdotted` style, except that the main entries just have the name displayed.

```
7538 \newglossarystyle{sublistdotted}{%
```

Base it on the listdotted style:

```
7539 \setglossarystyle{listdotted}%
```

Main (level 0) entries just display the name:

```
7540 \renewcommand*{\glossentry}[2]{%
7541 \item[\glssentryitem{##1}\glstarget{##1}{\glossentryname{##1}}}%
7542 }
```

5.4 Glossary Styles using longtable (the glossary-long package)

The glossary styles defined in the package used the longtable environment in the glossary.

```
7543 \ProvidesPackage{glossary-long}[2013/11/14 v4.0 (NLCT)]
```

Requires the package:

```
7544 \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`.)

```
7545 \@ifundefined{glsdescwidth}{%  
7546   \newlength{glsdescwidth}  
7547   \setlength{glsdescwidth}{0.6\hsize}  
7548 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column.

```
7549 \@ifundefined{glspagelistwidth}{%  
7550   \newlength{glspagelistwidth}  
7551   \setlength{glspagelistwidth}{0.1\hsize}  
7552 }{}
```

`long` The long glossary style command which uses the `longtable` environment:

```
7553 \newglossarystyle{long}{%
```

Use `longtable` with two columns:

```
7554   \renewenvironment{theglossary}%  
7555     {\begin{longtable}{lp{glsdescwidth}}}%  
7556     {\end{longtable}}%
```

Do nothing at the start of the environment:

```
7557   \renewcommand*{\glossaryheader}{}%
```

No heading between groups:

```
7558   \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries displayed in a row:

```
7559   \renewcommand{\glossentry}[2]{%  
7560     \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &  
7561     \glossentrydesc{##1}\glspostdescription\space ##2\tabularnewline  
7562   }%
```

Sub entries displayed on the following row without the name:

```
7563   \renewcommand{\subglossentry}[3]{%  
7564     &  
7565     \glssubentryitem{##2}%  
7566     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space  
7567     ##3\tabularnewline  
7568   }%
```

Blank row between groups:

```
7569   \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else &  
7570 \tabularnewline\fi}%  
7571 }
```

longborder The longborder style is like the above, but with horizontal and vertical lines:

```
7572 \newglossarystyle{longborder}{%
```

Base it on the glosstylelong style:

```
7573 \setglossarystyle{long}%
```

Use longtable with two columns with vertical lines between each column:

```
7574 \renewenvironment{theglossary}{%
```

```
7575 \begin{longtable}{|l|p{\glstdescwidth}|}{\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
7576 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
```

```
7577 }
```

longheader The longheader style is like the long style but with a header:

```
7578 \newglossarystyle{longheader}{%
```

Base it on the glosstylelong style:

```
7579 \setglossarystyle{long}%
```

Set the table's header:

```
7580 \renewcommand*{\glossaryheader}{%
```

```
7581 \bfseries \entryname & \bfseries \descriptionname\tabularnewline\endhead}%
```

```
7582 }
```

longheaderborder The longheaderborder style is like the long style but with a header and border:

```
7583 \newglossarystyle{longheaderborder}{%
```

Base it on the glosstylelongborder style:

```
7584 \setglossarystyle{longborder}%
```

Set the table's header and add horizontal line to table's foot:

```
7585 \renewcommand*{\glossaryheader}{%
```

```
7586 \hline\bfseries \entryname & \bfseries
```

```
7587 \descriptionname\tabularnewline\hline
```

```
7588 \endhead
```

```
7589 \hline\endfoot}%
```

```
7590 }
```

long3col The long3col style is like long but with 3 columns

```
7591 \newglossarystyle{long3col}{%
```

Use a longtable with 3 columns:

```
7592 \renewenvironment{theglossary}{%
```

```
7593 {\begin{longtable}{lp{\glstdescwidth}p{\glspagelistwidth}}}%
```

```
7594 {\end{longtable}}%
```

No table header:

```
7595 \renewcommand*{\glossaryheader}{}%
```

No headings between groups:

```
7596 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```

7597 \renewcommand{\glossentry}[2]{%
7598   \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7599   \glossentrydesc{##1} & ##2\tabularnewline
7600 }%
```

Sub-entries on a separate row (no name, description in second column, page list in third column):

```

7601 \renewcommand{\subglossentry}[3]{%
7602   &
7603   \glssubentryitem{##2}%
7604   \glstarget{##2}{\strut}\glossentrydesc{##2} &
7605   ##3\tabularnewline
7606 }%
```

Blank row between groups:

```

7607 \renewcommand*{\glsgroupskip}{%
7608   \ifglsgnogroupskip\else & &\tabularnewline\fi}%
7609 }
```

long3colborder The long3colborder style is like the long3col style but with a border:

```
7610 \newglossarystyle{long3colborder}{%
```

Base it on the glostylelong3col style:

```
7611 \setglossarystyle{long3col}%
```

Use a longtable with 3 columns with vertical lines around them:

```

7612 \renewenvironment{theglossary}%
7613   {\begin{longtable}{|l|p{\glsgdescwidth}|p{\glspagelistwidth}|}%
7614   {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```

7615 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
7616 }
```

long3colheader The long3colheader style is like long3col but with a header row:

```
7617 \newglossarystyle{long3colheader}{%
```

Base it on the glostylelong3col style:

```
7618 \setglossarystyle{long3col}%
```

Set the table's header:

```

7619 \renewcommand*{\glossaryheader}{%
7620   \bfseries\entryname&\bfseries\descriptionname&
7621   \bfseries\pagelistname\tabularnewline\endhead}%
7622 }
```

long3colheaderborder The long3colheaderborder style is like the above but with a border

```
7623 \newglossarystyle{long3colheaderborder}{%
```

Base it on the `glostylelong3colborder` style:

```
7624 \setglossarystyle{long3colborder}{%
```

Set the table's header and add horizontal line at table's foot:

```
7625 \renewcommand*{\glossaryheader}{{%  
7626 \hline  
7627 \bfseries\entryname&\bfseries\descriptionname&  
7628 \bfseries\pagelistname\tabularnewline\hline\endhead  
7629 \hline\endfoot}%  
7630 }
```

`long4col` The `long4col` style has four columns where the third column contains the value of the associated symbol key.

```
7631 \newglossarystyle{long4col}{%
```

Use a `longtable` with 4 columns:

```
7632 \renewenvironment{theglossary}{%  
7633 {\begin{longtable}{llll}}%  
7634 {\end{longtable}}%
```

No table header:

```
7635 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7636 \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):

```
7637 \renewcommand{\glossentry}[2]{%  
7638 \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &  
7639 \glossentrydesc{##1} &  
7640 \glossentrysymbol{##1} &  
7641 ##2\tabularnewline  
7642 }%
```

Sub entries on a single row with no name (description in second column, symbol in third column, page list in last column):

```
7643 \renewcommand{\subglossentry}[3]{%  
7644 &  
7645 \glssubentryitem{##2}%  
7646 \glstarget{##2}{\strut}\glossentrydesc{##2} &  
7647 \glossentrysymbol{##2} & ##3\tabularnewline  
7648 }%
```

Blank row between groups:

```
7649 \renewcommand*{\glsgroupskip}{%  
7650 \ifglsgnোগroupskip\else & & \tabularnewline\fi}%  
7651 }
```

`long4colheader` The `long4colheader` style is like `long4col` but with a header row.

```
7652 \newglossarystyle{long4colheader}{%
```


Base it on the `glostylelong4col` style:

```
7653 \setglossarystyle{long4col}%
```

Table has a header:

```
7654 \renewcommand*{\glossaryheader}{%  
7655 \bfseries\entryname&\bfseries\descriptionname&  
7656 \bfseries \symbolname&  
7657 \bfseries\pagelistname\tabularnewline\endhead}%  
7658 }
```

`long4colborder` The `long4colborder` style is like `long4col` but with a border.

```
7659 \newglossarystyle{long4colborder}{%
```

Base it on the `glostylelong4col` style:

```
7660 \setglossarystyle{long4col}%
```

Use a `longtable` with 4 columns surrounded by vertical lines:

```
7661 \renewenvironment{theglossary}%  
7662 {\begin{longtable}{|l|l|l|l|}}%  
7663 {\end{longtable}}%
```

Add horizontal lines to the head and foot of the table:

```
7664 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%  
7665 }
```

`long4colheaderborder` The `long4colheaderborder` style is like the above but with a border.

```
7666 \newglossarystyle{long4colheaderborder}{%
```

Base it on the `glostylelong4col` style:

```
7667 \setglossarystyle{long4col}%
```

Use a `longtable` with 4 columns surrounded by vertical lines:

```
7668 \renewenvironment{theglossary}%  
7669 {\begin{longtable}{|l|l|l|l|}}%  
7670 {\end{longtable}}%
```

Add table header and horizontal line at the table's foot:

```
7671 \renewcommand*{\glossaryheader}{%  
7672 \hline\bfseries\entryname&\bfseries\descriptionname&  
7673 \bfseries \symbolname&  
7674 \bfseries\pagelistname\tabularnewline\hline\endhead  
7675 \hline\endfoot}%  
7676 }
```

`altlong4col` The `altlong4col` style is like the `long4col` style but can have multiline descriptions and page lists.

```
7677 \newglossarystyle{altlong4col}{%
```

Base it on the `glostylelong4col` style:

```
7678 \setglossarystyle{long4col}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
7679 \renewenvironment{theglossary}%
7680   {\begin{longtable}{lp{\glsdescwidth}lp{\glspagelistwidth}}}%
7681   {\end{longtable}}%
7682 }
```

altlong4colheader The altlong4colheader style is like altlong4col but with a header row.

```
7683 \newglossarystyle{altlong4colheader}{%
```

Base it on the glostylelong4colheader style:

```
7684 \setglossarystyle{long4colheader}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
7685 \renewenvironment{theglossary}%
7686   {\begin{longtable}{lp{\glsdescwidth}lp{\glspagelistwidth}}}%
7687   {\end{longtable}}%
7688 }
```

altlong4colborder The altlong4colborder style is like altlong4col but with a border.

```
7689 \newglossarystyle{altlong4colborder}{%
```

Base it on the glostylelong4colborder style:

```
7690 \setglossarystyle{long4colborder}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
7691 \renewenvironment{theglossary}%
7692   {\begin{longtable}{|lp{\glsdescwidth}|lp{\glspagelistwidth}|}}%
7693   {\end{longtable}}%
7694 }
```

ong4colheaderborder The altlong4colheaderborder style is like the above but with a header as well as a border.

```
7695 \newglossarystyle{altlong4colheaderborder}{%
```

Base it on the glostylelong4colheaderborder style:

```
7696 \setglossarystyle{long4colheaderborder}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
7697 \renewenvironment{theglossary}%
7698   {\begin{longtable}{|lp{\glsdescwidth}|lp{\glspagelistwidth}|}}%
7699   {\end{longtable}}%
7700 }
```

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.

7701 \ProvidesPackage{glossary-longragged}[2014/07/30 v4.08 (NLCT)]

Requires the package:

7702 \RequirePackage{array}

Requires the package:

7703 \RequirePackage{longtable}

`\glsdescwidth` This is a length that governs the width of the description column. This may have already been defined.

7704 \@ifundefined{glsdescwidth}{%

7705 \newlength{glsdescwidth

7706 \setlength{glsdescwidth}{0.6\hsize}

7707 }{}

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

7708 \@ifundefined{glspagelistwidth}{%

7709 \newlength{glspagelistwidth

7710 \setlength{glspagelistwidth}{0.1\hsize}

7711 }{}

`longragged` The longragged glossary style is like the long but uses ragged right formatting for the description column.

7712 \newglossarystyle{longragged}{%

Use longtable with two columns:

7713 \renewenvironment{theglossary}{%

7714 {\begin{longtable}{l>{\raggedright}p{glsdescwidth}}}%

7715 {\end{longtable}}}%

Do nothing at the start of the environment:

7716 \renewcommand*{\glossaryheader}{}%

No heading between groups:

7717 \renewcommand*{\glsgroupheading}[1]{}%

Main (level 0) entries displayed in a row:

7718 \renewcommand{\glossentry}[2]{%

7719 \glstarget{##1}{\glossentryname{##1}} &

7720 \glossentrydesc{##1}\glspostdescription\space ##2%

7721 \tabularnewline

7722 }%

Sub entries displayed on the following row without the name:

```

7723 \renewcommand{\subglossentry}[3]{%
7724     &
7725     \glssubentryitem{##2}%
7726     \glstarget{##2}{\strut}\glossentrydesc{##2}%
7727     \glspostdescription\space ##3%
7728     \tabularnewline
7729 }%
```

Blank row between groups:

```

7730 \renewcommand*{\glsgroupskip}{\ifglsgnogroupskip\else & \tabularnewline\fi}%
7731 }
```

longraggedborder The longraggedborder style is like the above, but with horizontal and vertical lines:

```

7732 \newglossarystyle{longraggedborder}{%
```

Base it on the glostylelongragged style:

```

7733 \setglossarystyle{longragged}%
```

Use longtable with two columns with vertical lines between each column:

```

7734 \renewenvironment{theglossary}{%
7735     \begin{longtable}{|l|>{\raggedright}p{\glsglscwidth}|}%
7736     {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```

7737 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
7738 }
```

longraggedheader The longraggedheader style is like the longragged style but with a header:

```

7739 \newglossarystyle{longraggedheader}{%
```

Base it on the glostylelongragged style:

```

7740 \setglossarystyle{longragged}%
```

Set the table's header:

```

7741 \renewcommand*{\glossaryheader}{%
7742     \bfseries \entryname & \bfseries \descriptionname
7743     \tabularnewline\endhead}%
7744 }
```

longraggedheaderborder The longraggedheaderborder style is like the longragged style but with a header and border:

```

7745 \newglossarystyle{longraggedheaderborder}{%
```

Base it on the glostylelongraggedborder style:

```

7746 \setglossarystyle{longraggedborder}%
```

Set the table's header and add horizontal line to table's foot:

```

7747 \renewcommand*{\glossaryheader}{%
7748     \hline\bfseries \entryname & \bfseries \descriptionname
```

```

7749 \tabularnewline\hline
7750 \endhead
7751 \hline\endfoot}%
7752 }

```

`longragged3col` The `longragged3col` style is like `longragged` but with 3 columns

```

7753 \newglossarystyle{longragged3col}{%

```

Use a longtable with 3 columns:

```

7754 \renewenvironment{theglossary}%
7755 {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}%
7756 >{\raggedright}p{\glspagelistwidth}}}%
7757 {\end{longtable}}%

```

No table header:

```

7758 \renewcommand*{\glossaryheader}{}%

```

No headings between groups:

```

7759 \renewcommand*{\glsgroupheading}[1]{}%

```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```

7760 \renewcommand{\glossentry}[2]{%
7761 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7762 \glossentrydesc{##1} & ##2\tabularnewline
7763 }%

```

Sub-entries on a separate row (no name, description in second column, page list in third column):

```

7764 \renewcommand{\subglossentry}[3]{%
7765 &
7766 \glssubentryitem{##2}%
7767 \glstarget{##2}{\strut}\glossentrydesc{##2} &
7768 ##3\tabularnewline
7769 }%

```

Blank row between groups:

```

7770 \renewcommand*{\glsgroupskip}{%
7771 \ifglsnogroupskip\else & &\tabularnewline\fi}%
7772 }

```

`longragged3colborder` The `longragged3colborder` style is like the `longragged3col` style but with a border:

```

7773 \newglossarystyle{longragged3colborder}{%

```

Base it on the `glostylelongragged3col` style:

```

7774 \setglossarystyle{longragged3col}%

```

Use a longtable with 3 columns with vertical lines around them:

```

7775 \renewenvironment{theglossary}%
7776 {\begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|}%
7777 >{\raggedright}p{\glspagelistwidth}|}%
7778 {\end{longtable}}%

```

Place horizontal lines at the head and foot of the table:

```
7779 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
7780 }
```

`longragged3colheader` The `longragged3colheader` style is like `longragged3col` but with a header row:

```
7781 \newglossarystyle{longragged3colheader}{%
```

Base it on the `glostylelongragged3col` style:

```
7782 \setglossarystyle{longragged3col}%
```

Set the table's header:

```
7783 \renewcommand*{\glossaryheader}{%
7784 \bfseries\entryname&\bfseries\descriptionname&
7785 \bfseries\pagelistname\tabularnewline\endhead}%
7786 }
```

`longragged3colheaderborder` The `longragged3colheaderborder` style is like the above but with a border

```
7787 \newglossarystyle{longragged3colheaderborder}{%
```

Base it on the `glostylelongragged3colborder` style:

```
7788 \setglossarystyle{longragged3colborder}%
```

Set the table's header and add horizontal line at table's foot:

```
7789 \renewcommand*{\glossaryheader}{%
7790 \hline
7791 \bfseries\entryname&\bfseries\descriptionname&
7792 \bfseries\pagelistname\tabularnewline\hline\endhead
7793 \hline\endfoot}%
7794 }
```

`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.

```
7795 \newglossarystyle{altlongragged4col}{%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
7796 \renewenvironment{theglossary}%
7797 {\begin{longtable}{l>{\raggedright}p{\glstdescwidth}l%
7798 >{\raggedright}p{\glspagelistwidth}}}%
7799 {\end{longtable}}%
```

No table header:

```
7800 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7801 \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):

```
7802 \renewcommand{\glossentry}[2]{}%
```

```

7803   \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7804   \glossentrydesc{##1} & \glossentrysymbol{##1} &
7805   ##2\tabularnewline
7806 }%

```

Sub entries on a single row with no name (description in second column, symbol in third column, page list in last column):

```

7807   \renewcommand{\subglossentry}[3]{%
7808     &
7809     \glssubentryitem{##2}%
7810     \glstarget{##2}{\strut}\glossentrydesc{##2} &
7811     \glossentrysymbol{##2} & ##3\tabularnewline
7812 }%

```

Blank row between groups:

```

7813   \renewcommand*\glsgroupskip{%
7814     \ifglsgroupskip\else & & \tabularnewline\fi}%
7815 }

```

`longragged4colheader` The `altlongragged4colheader` style is like `altlongragged4col` but with a header row.

```

7816 \newglossarystyle{altlongragged4colheader}{%

```

Base it on the `glostylealtlongragged4col` style:

```

7817   \setglossarystyle{altlongragged4col}%

```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```

7818   \renewenvironment{theglossary}%
7819     {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%
7820       >{\raggedright}p{\glspagelistwidth}}}%
7821     {\end{longtable}}%

```

Table has a header:

```

7822   \renewcommand*\glossaryheader{%
7823     \bfseries\entryname&\bfseries\descriptionname&
7824     \bfseries \symbolname&
7825     \bfseries\pagelistname\tabularnewline\endhead}%
7826 }

```

`longragged4colborder` The `altlongragged4colborder` style is like `altlongragged4col` but with a border.

```

7827 \newglossarystyle{altlongragged4colborder}{%

```

Base it on the `glostylealtlongragged4col` style:

```

7828   \setglossarystyle{altlongragged4col}%

```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```

7829   \renewenvironment{theglossary}%
7830     {\begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|l|%
7831       >{\raggedright}p{\glspagelistwidth}|}%
7832     {\end{longtable}}%

```

Add horizontal lines to the head and foot of the table:

```
7833 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
7834 }
```

`ged4colheaderborder` The `altlongragged4colheaderborder` style is like the above but with a header as well as a border.

```
7835 \newglossarystyle{altlongragged4colheaderborder}{%
```

Base it on the `glostylealtlongragged4col` style:

```
7836 \setglossarystyle{altlongragged4col}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
7837 \renewenvironment{theglossary}%
7838 {\begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|l|}%
7839 >{\raggedright}p{\glspagelistwidth}|}%
7840 {\end{longtable}}%
```

Add table header and horizontal line at the table's foot:

```
7841 \renewcommand*{\glossaryheader}{%
7842 \hline\bfseries\entryname&\bfseries\descriptionname&
7843 \bfseries \symbolname&
7844 \bfseries\pagelistname\tabularnewline\hline\endhead
7845 \hline\endfoot}%
7846 }
```

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.

```
7847 \ProvidesPackage{glossary-mcols}[2013/11/14 v4.0 (NLCT)]
```

Required packages:

```
7848 \RequirePackage{multicol}
7849 \RequirePackage{glossary-tree}
```

`\glscols` Define macro in which to store the number of columns. (Defaults to 2.)

```
7850 \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.)

```
7851 \newglossarystyle{mcolindex}{%
7852 \setglossarystyle{index}%
7853 \renewenvironment{theglossary}%
7854 {%
```



```

7855 \begin{multicols}{\glsmcols}
7856 \setlength{\parindent}{0pt}%
7857 \setlength{\parskip}{0pt plus 0.3pt}%
7858 \let\item\@idxitem}%
7859 {\end{multicols}}%
7860 }

```

mcolindexgroup As mcolindex but has headings:

```

7861 \newglossarystyle{mcolindexgroup}{%
7862 \setglossarystyle{mcolindex}%
7863 \renewcommand*{\glsgroupheading}[1]{%
7864 \item\textbf{\glsgrouptitle{##1}}\indexspace}%
7865 }

```

mcolindexhypergroup The mcolindexhypergroup style is like the mcolindexgroup style but has hyper navigation.

```

7866 \newglossarystyle{mcolindexhypergroup}{%
  Base it on the glostylemcolindex style:
7867 \setglossarystyle{mcolindex}%
  Put navigation links to the groups at the start of the glossary:
7868 \renewcommand*{\glossaryheader}{%
7869 \item\textbf{\glsnavigation}\indexspace}%
  Add a heading for each group (with a target). The group's title is in bold followed
  by a vertical gap.
7870 \renewcommand*{\glsgroupheading}[1]{%
7871 \item\textbf{\glsnavhypertarget{##1}}{\glsgrouptitle{##1}}}%
7872 \indexspace}%
7873 }

```

mcoltree Multi-column index style. Same as the tree, but puts the glossary in multiple columns.

```

7874 \newglossarystyle{mcoltree}{%
7875 \setglossarystyle{tree}%
7876 \renewenvironment{theglossary}%
7877 {%
7878 \begin{multicols}{\glsmcols}
7879 \setlength{\parindent}{0pt}%
7880 \setlength{\parskip}{0pt plus 0.3pt}%
7881 }%
7882 {\end{multicols}}%
7883 }

```

mcoltreegroup Like the mcoltree style but the glossary groups have headings.

```

7884 \newglossarystyle{mcoltreegroup}{%
  Base it on the glostylemcoltree style:
7885 \setglossarystyle{mcoltree}%

```

Each group has a heading (in bold) followed by a vertical gap):

```
7886 \renewcommand{\glsgroupheading}[1]{\par
7887 \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
7888 }
```

mcoltreehypergroup The mcoltreehypergroup style is like the treegroup style, but has a set of links to the groups at the start of the glossary.

```
7889 \newglossarystyle{mcoltreehypergroup}{%
```

Base it on the glostylemcoltree style:

```
7890 \setglossarystyle{mcoltree}%
```

Put navigation links to the groups at the start of the theglossary environment:

```
7891 \renewcommand*{\glossaryheader}{%
7892 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
7893 \renewcommand*{\glsgroupheading}[1]{%
7894 \par\noindent
7895 \textbf{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}\par
7896 \indexspace}%
7897 }
```

mcoltreenoname Multi-column index style. Same as the treenoname, but puts the glossary in multiple columns.

```
7898 \newglossarystyle{mcoltreenoname}{%
7899 \setglossarystyle{treenoname}%
7900 \renewenvironment{theglossary}%
7901 {%
7902 \begin{multicols}{\glsmcols}
7903 \setlength{\parindent}{0pt}%
7904 \setlength{\parskip}{0pt plus 0.3pt}%
7905 }%
7906 {\end{multicols}}}%
7907 }
```

mcoltreenonamegroup Like the mcoltreenoname style but the glossary groups have headings.

```
7908 \newglossarystyle{mcoltreenonamegroup}{%
```

Base it on the glostylemcoltreenoname style:

```
7909 \setglossarystyle{mcoltreenoname}%
```

Give each group a heading:

```
7910 \renewcommand{\glsgroupheading}[1]{\par
7911 \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
7912 }
```

treenonamehypergroup The mcoltreenonamehypergroup style is like the mcoltreenonamegroup style, but has a set of links to the groups at the start of the glossary.

```
7913 \newglossarystyle{mcoltreenonamehypergroup}{%
```

Base it on the `glostylemcoltreenoname` style:

```
7914 \setglossarystyle{mcoltreenoname}%
```

Put navigation links to the groups at the start of the `theglossary` environment:

```
7915 \renewcommand*{\glossaryheader}{%
```

```
7916 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
7917 \renewcommand*{\glsgroupheading}[1]{%
```

```
7918 \par\noindent
```

```
7919 \textbf{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}\par
```

```
7920 \indexspace}%
```

```
7921 }
```

`mcolalmtree` Multi-column index style. Same as the `almtree`, but puts the glossary in multiple columns.

```
7922 \newglossarystyle{mcolalmtree}{%
```

```
7923 \setglossarystyle{almtree}%
```

```
7924 \renewenvironment{theglossary}%
```

```
7925 {%
```

```
7926 \begin{multicols}{\glsmcols}
```

```
7927 \def\@gls@prevlevel{-1}%
```

```
7928 \mbox{}\par
```

```
7929 }%
```

```
7930 {\par\end{multicols}}%
```

```
7931 }
```

`mcolalmtreegroup` Like the `mcolalmtree` style but the glossary groups have headings.

```
7932 \newglossarystyle{mcolalmtreegroup}{%
```

Base it on the `glostylemcolalmtree` style:

```
7933 \setglossarystyle{mcolalmtree}%
```

Give each group a heading.

```
7934 \renewcommand{\glsgroupheading}[1]{\par
```

```
7935 \def\@gls@prevlevel{-1}%
```

```
7936 \hangindent0pt\relax
```

```
7937 \parindent0pt\relax
```

```
7938 \textbf{\glsgrouptitle{##1}}\par\indexspace}%
```

```
7939 }
```

`olalmtreehypergroup` The `mcolalmtreehypergroup` style is like the `mcolalmtreegroup` style, but has a set of links to the groups at the start of the glossary.

```
7940 \newglossarystyle{mcolalmtreehypergroup}{%
```

Base it on the `glostylemcolalmtree` style:

```
7941 \setglossarystyle{mcolalmtree}%
```

Put the navigation links in the header

```
7942 \renewcommand*{\glossaryheader}{%
7943   \par
7944   \def\@gls@prevlevel{-1}%
7945   \hangindent0pt\relax
7946   \parindent0pt\relax
7947   \textbf{\glsnavigation}\par\indexspace}%
```

Put a hypertarget at the start of each group

```
7948 \renewcommand*{\glsgroupheading}[1]{%
7949   \par
7950   \def\@gls@prevlevel{-1}%
7951   \hangindent0pt\relax
7952   \parindent0pt\relax
7953   \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
7954   \indexspace}}
```

5.7 Glossary Styles using supertabular environment (glossary-super package)

The glossary styles defined in the package use the supertabular environment.

```
7955 \ProvidesPackage{glossary-super}[2013/11/14 v4.0 (NLCT)]
```

Requires the package:

```
7956 \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.

```
7957 \@ifundefined{glsdescwidth}{%
7958   \newlength\glsdescwidth
7959   \setlength{\glsdescwidth}{0.6\hsize}
7960 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined if has been loaded.

```
7961 \@ifundefined{glspagelistwidth}{%
7962   \newlength\glspagelistwidth
7963   \setlength{\glspagelistwidth}{0.1\hsize}
7964 }{}
```

`super` The super glossary style uses the supertabular environment (it uses lengths defined in the package.)

```
7965 \newglossarystyle{super}{%
```

Put the glossary in a supertabular environment with two columns and no head or tail:

```
7966   \renewenvironment{theglossary}{%
7967     {\tablehead{}\tabletail}%
7968     \begin{supertabular}{lp{\glsdescwidth}}%
7969     {\end{supertabular}}%
```

Do nothing at the start of the table:

```
7970 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7971 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries put in a row (name in first column, description and page list in second column):

```
7972 \renewcommand{\glossentry}[2]{%
7973   \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7974   \glossentrydesc{##1}\glspostdescription\space ##2\tabularnewline
7975 }
```

Sub entries put in a row (no name, description and page list in second column):

```
7976 \renewcommand{\subglossentry}[3]{%
7977   &
7978   \glssubentryitem{##2}%
7979   \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space
7980   ##3\tabularnewline
7981 }
```

Blank row between groups:

```
7982 \renewcommand*{\glsgroupskip}{}%
7983 \ifglsgnognroupskip\else & \tabularnewline\fi}%
7984 }
```

superborder The superborder style is like the above, but with horizontal and vertical lines:

```
7985 \newglossarystyle{superborder}{}%
```

Base it on the `glostylesuper` style:

```
7986 \setglossarystyle{super}{}%
```

Put the glossary in a `supertabular` environment with two columns and a horizontal line in the head and tail:

```
7987 \renewenvironment{theglossary}%
7988   {\tablehead{\hline}\tabletail{\hline}}%
7989   \begin{supertabular}{|l|p{\glsdescwidth}|}%
7990   {\end{supertabular}}%
7991 }
```

superheader The superheader style is like the super style, but with a header:

```
7992 \newglossarystyle{superheader}{}%
```

Base it on the `glostylesuper` style:

```
7993 \setglossarystyle{super}{}%
```

Put the glossary in a `supertabular` environment with two columns, a header and no tail:

```
7994 \renewenvironment{theglossary}%
7995   {\tablehead{\bfseries \entryname &
```

```

7996 \bfseries\descriptionname\tabularnewline}%
7997 \tabletail{}%
7998 \begin{supertabular}{lp{\glsgdescwidth}}}%
7999 {\end{supertabular}}%
8000 }

```

superheaderborder The superheaderborder style is like the super style but with a header and border:

```
8001 \newglossarystyle{superheaderborder}{%
```

Base it on the glostylesuper style:

```
8002 \setglossarystyle{super}%
```

Put the glossary in a supertabular environment with two columns, a header and horizontal lines above and below the table:

```

8003 \renewenvironment{theglossary}%
8004 {\tablehead{\hline\bfseries \entryname &
8005 \bfseries \descriptionname\tabularnewline\hline}%
8006 \tabletail{\hline}
8007 \begin{supertabular}{|lp{\glsgdescwidth}|}%
8008 {\end{supertabular}}%
8009 }

```

super3col The super3col style is like the super style, but with 3 columns:

```
8010 \newglossarystyle{super3col}{%
```

Put the glossary in a supertabular environment with three columns and no head or tail:

```

8011 \renewenvironment{theglossary}%
8012 {\tablehead{}\tabletail{}%
8013 \begin{supertabular}{lp{\glsgdescwidth}p{\glspagelistwidth}}}%
8014 {\end{supertabular}}%

```

Do nothing at the start of the table:

```
8015 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
8016 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```

8017 \renewcommand{\glossentry}[2]{%
8018 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
8019 \glossentrydesc{##1} & ##2\tabularnewline
8020 }%

```

Sub entries on a row (no name, description in second column, page list in last column):

```

8021 \renewcommand{\subglossentry}[3]{%
8022 &
8023 \glssubentryitem{##2}%
8024 \glstarget{##2}{\strut}\glossentrydesc{##2} &

```

```
8025      ##3\tabularnewline
```

```
8026  }%
```

Blank row between groups:

```
8027  \renewcommand*{\glsgroupskip}{%
```

```
8028    \ifglsgnোগroupskip\else & &\tabularnewline\fi}%
```

```
8029 }
```

super3colborder The **super3colborder** style is like the **super3col** style, but with a border:

```
8030 \newglossarystyle{super3colborder}{%
```

Base it on the **glostylessuper3col** style:

```
8031  \setglossarystyle{super3col}%
```

Put the glossary in a **supertabular** environment with three columns and a horizontal line in the head and tail:

```
8032  \renewenvironment{theglossary}%
```

```
8033    {\tablehead{\hline}\tabletail{\hline}%
```

```
8034      \begin{supertabular}{|l|p{\glsgdescwidth}|p{\glspagelistwidth|}}%
```

```
8035      {\end{supertabular}}%
```

```
8036 }
```

super3colheader The **super3colheader** style is like the **super3col** style but with a header row:

```
8037 \newglossarystyle{super3colheader}{%
```

Base it on the **glostylessuper3col** style:

```
8038  \setglossarystyle{super3col}%
```

Put the glossary in a **supertabular** environment with three columns, a header and no tail:

```
8039  \renewenvironment{theglossary}%
```

```
8040    {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
```

```
8041      \bfseries\pagelistname\tabularnewline}\tabletail{}}%
```

```
8042      \begin{supertabular}{lp{\glsgdescwidth}p{\glspagelistwidth}}%
```

```
8043      {\end{supertabular}}%
```

```
8044 }
```

super3colheaderborder The **super3colheaderborder** style is like the **super3col** style but with a header and border:

```
8045 \newglossarystyle{super3colheaderborder}{%
```

Base it on the **glostylessuper3colborder** style:

```
8046  \setglossarystyle{super3colborder}%
```

Put the glossary in a **supertabular** environment with three columns, a header with horizontal lines and a horizontal line in the tail:

```
8047  \renewenvironment{theglossary}%
```

```
8048    {\tablehead{\hline
```

```
8049      \bfseries\entryname&\bfseries\descriptionname&
```

```
8050      \bfseries\pagelistname\tabularnewline\hline}%
```

```
8051      \tabletail{\hline}%
```

```

8052     \begin{supertabular}{|l|p{\glstdescwidth}|p{\glspagelistwidth}|}%
8053     {\end{supertabular}}%
8054 }

```

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.

```
8055 \newglossarystyle{super4col}{%
```

Put the glossary in a `supertabular` environment with four columns and no head or tail:

```

8056   \renewenvironment{theglossary}%
8057   {\tablehead{}\tabletail}%
8058   \begin{supertabular}{|l|l|l|l|}%
8059   \end{supertabular}}%

```

Do nothing at the start of the table:

```
8060   \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
8061   \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:

```

8062   \renewcommand{\glossentry}[2]{%
8063     \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
8064     \glossentrydesc{##1} &
8065     \glossentrysymbol{##1} & ##3\tabularnewline
8066   }%

```

Sub entries on a row with no name, the description in the second column, symbol in third column and page list in last column:

```

8067   \renewcommand{\subglossentry}[3]{%
8068     &
8069     \glssubentryitem{##2}%
8070     \glstarget{##2}{\strut}\glossentrydesc{##2} &
8071     \glossentrysymbol{##2} & ##3\tabularnewline
8072   }%

```

Blank row between groups:

```

8073   \renewcommand*{\glsgroupskip}{%
8074     \ifglsnogroupskip\else & & \tabularnewline\fi}%
8075 }

```

super4colheader The `super4colheader` style is like the `super4col` but with a header row.

```
8076 \newglossarystyle{super4colheader}{%
```

Base it on the `glostylessuper4col` style:

```
8077   \setglossarystyle{super4col}%
```

Put the glossary in a `supertabular` environment with four columns, a header and no tail:


```

8078 \renewenvironment{theglossary}%
8079   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
8080     \bfseries\symbolname &
8081     \bfseries\pagelistname\tabularnewline}%
8082   \tabletail{}}%
8083   \begin{supertabular}{|l|l|l|l|}%
8084   {\end{supertabular}}}%
8085 }

```

super4colborder The super4colborder style is like the super4col but with a border.

```

8086 \newglossarystyle{super4colborder}{%
  Base it on the glostylesuper4col style:
8087 \setglossarystyle{super4col}%
  Put the glossary in a supertabular environment with four columns and a horizontal line in the head and tail:
8088 \renewenvironment{theglossary}%
8089   {\tablehead{\hline}\tabletail{\hline}%
8090   \begin{supertabular}{|l|l|l|l|}%
8091   {\end{supertabular}}}%
8092 }

```

per4colheaderborder The super4colheaderborder style is like the super4col but with a header and border.

```

8093 \newglossarystyle{super4colheaderborder}{%
  Base it on the glostylesuper4col style:
8094 \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:
8095 \renewenvironment{theglossary}%
8096   {\tablehead{\hline\bfseries\entryname&\bfseries\descriptionname&
8097     \bfseries\symbolname &
8098     \bfseries\pagelistname\tabularnewline\hline}%
8099   \tabletail{\hline}%
8100   \begin{supertabular}{|l|l|l|l|}%
8101   {\end{supertabular}}}%
8102 }

```

altsuper4col The altsuper4col glossary style is like super4col but has provision for multiline descriptions.

```

8103 \newglossarystyle{altsuper4col}{%
  Base it on the glostylesuper4col style:
8104 \setglossarystyle{super4col}%
  Put the glossary in a supertabular environment with four columns and no head or tail:

```

```

8105 \renewenvironment{theglossary}%
8106   {\tablehead{}\tabletail{}}%
8107   \begin{supertabular}{lp{\glsgdescwidth}lp{\glspagelistwidth}}}%
8108   {\end{supertabular}}}%
8109 }

```

altsuper4colheader The altsuper4colheader style is like the altsuper4col but with a header row.

```

8110 \newglossarystyle{altsuper4colheader}{%
      Base it on the glostylesuper4colheader style:
8111   \setglossarystyle{super4colheader}%
      Put the glossary in a supertabular environment with four columns, a header and
      no tail:
8112   \renewenvironment{theglossary}%
8113     {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
8114               \bfseries\symbolname &
8115               \bfseries\pagelistname\tablearnewline}\tabletail{}}%
8116     \begin{supertabular}{lp{\glsgdescwidth}lp{\glspagelistwidth}}}%
8117     {\end{supertabular}}}%
8118 }

```

altsuper4colborder The altsuper4colborder style is like the altsuper4col but with a border.

```

8119 \newglossarystyle{altsuper4colborder}{%
      Base it on the glostylesuper4colborder style:
8120   \setglossarystyle{super4colborder}%
      Put the glossary in a supertabular environment with four columns and a hori-
      zontal line in the head and tail:
8121   \renewenvironment{theglossary}%
8122     {\tablehead{\hline}\tabletail{\hline}%
8123     \begin{supertabular}%
8124       {\lllp{\glsgdescwidth}lllp{\glspagelistwidth}ll}%
8125     {\end{supertabular}}}%
8126 }

```

per4colheaderborder The altsuper4colheaderborder style is like the altsuper4col but with a header and border.

```

8127 \newglossarystyle{altsuper4colheaderborder}{%
      Base it on the glostylesuper4colheaderborder style:
8128   \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:
8129   \renewenvironment{theglossary}%
8130     {\tablehead{\hline
8131               \bfseries\entryname &
8132               \bfseries\descriptionname &
8133               \bfseries\symbolname &

```

```

8134      \bfseries\pagelistname\tabularnewline\hline}%
8135      \tabletail{\hline}%
8136      \begin{supertabular}%
8137        {\lllp{\glsdescwidth}\lllp{\glspagelistwidth}}}%
8138      {\end{supertabular}}%
8139 }

```

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.

```
8140 \ProvidesPackage{glossary-superragged}[2013/11/14 v4.0 (NLCT)]
```

Requires the package:

```
8141 \RequirePackage{array}
```

Requires the package:

```
8142 \RequirePackage{supertabular}
```

`\glsdescwidth` This is a length that governs the width of the description column. This may already have been defined.

```

8143 \@ifundefined{glsdescwidth}{%
8144   \newlength\glsdescwidth
8145   \setlength{\glsdescwidth}{0.6\hsize}
8146 }{}

```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

```

8147 \@ifundefined{glspagelistwidth}{%
8148   \newlength\glspagelistwidth
8149   \setlength{\glspagelistwidth}{0.1\hsize}
8150 }{}

```

`superragged` The superragged glossary style uses the supertabular environment.

```
8151 \newglossarystyle{superragged}{%
```

Put the glossary in a supertabular environment with two columns and no head or tail:

```

8152   \renewenvironment{theglossary}%
8153     {\tablehead{}\tabletail{}}%
8154     \begin{supertabular}{l>{\raggedright}p{\glsdescwidth}}}%
8155     {\end{supertabular}}%

```

Do nothing at the start of the table:

```
8156   \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
8157   \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries put in a row (name in first column, description and page list in second column):

```

8158 \renewcommand{\glossentry}[2]{%
8159   \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
8160   \glossentrydesc{##1}\glspostdescription\space ##2%
8161   \tabularnewline
8162 }%

```

Sub entries put in a row (no name, description and page list in second column):

```

8163 \renewcommand{\subglossentry}[3]{%
8164   &
8165   \glssubentryitem{##2}%
8166   \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space
8167   ##3%
8168   \tabularnewline
8169 }%

```

Blank row between groups:

```

8170 \renewcommand*{\glsgroupskip}{\ifglsgnোগroupskip\else & \tabularnewline\fi}%
8171 }

```

superraggedborder The superraggedborder style is like the above, but with horizontal and vertical lines:

```

8172 \newglossarystyle{superraggedborder}{%

```

Base it on the glostylessuperragged style:

```

8173 \setglossarystyle{superragged}%

```

Put the glossary in a supertabular environment with two columns and a horizontal line in the head and tail:

```

8174 \renewenvironment{theglossary}%
8175   {\tablehead{\hline}\tabletail{\hline}%
8176   \begin{supertabular}{|l|>\raggedright}p{\glsgdescwidth}}}%
8177   {\end{supertabular}}%
8178 }

```

superraggedheader The superraggedheader style is like the super style, but with a header:

```

8179 \newglossarystyle{superraggedheader}{%

```

Base it on the glostylessuperragged style:

```

8180 \setglossarystyle{superragged}%

```

Put the glossary in a supertabular environment with two columns, a header and no tail:

```

8181 \renewenvironment{theglossary}%
8182   {\tablehead{\bfseries \entryname & \bfseries \descriptionname
8183   \tabularnewline}%
8184   \tabletail{}}%
8185   \begin{supertabular}{|l|>\raggedright}p{\glsgdescwidth}}}%
8186   {\end{supertabular}}%
8187 }

```

rraggedheaderborder The superraggedheaderborder style is like the superragged style but with a header and border:

```
8188 \newglossarystyle{superraggedheaderborder}{%
```

Base it on the glostylesuper style:

```
8189 \setglossarystyle{superragged}%
```

Put the glossary in a supertabular environment with two columns, a header and horizontal lines above and below the table:

```
8190 \renewenvironment{theglossary}%  
8191 {\tablehead{\hline\bfseries \entryname &  
8192 \bfseries \descriptionname\tabularnewline\hline}%  
8193 \tabletail{\hline}  
8194 \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}|}%  
8195 {\end{supertabular}}%  
8196 }
```

superragged3col The superragged3col style is like the superragged style, but with 3 columns:

```
8197 \newglossarystyle{superragged3col}{%
```

Put the glossary in a supertabular environment with three columns and no head or tail:

```
8198 \renewenvironment{theglossary}%  
8199 {\tablehead{}\tabletail{}}%  
8200 \begin{supertabular}{|l>{\raggedright}p{\glsgdescwidth}%  
8201 >{\raggedright}p{\glspagelistwidth}}}%  
8202 {\end{supertabular}}%
```

Do nothing at the start of the table:

```
8203 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
8204 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
8205 \renewcommand{\glossentry}[2]{%  
8206 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &  
8207 \glossentrydesc{##1} &  
8208 ##2\tabularnewline  
8209 }%
```

Sub entries on a row (no name, description in second column, page list in last column):

```
8210 \renewcommand{\subglossentry}[3]{%  
8211 &  
8212 \glssubentryitem{##2}%  
8213 \glstarget{##2}{\strut}\glossentrydesc{##2} &  
8214 ##3\tabularnewline  
8215 }%
```

Blank row between groups:

```
8216 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & \tabularnewline\fi}%  
8217 }
```

superragged3colborder The `superragged3colborder` style is like the `superragged3col` style, but with a border:

```
8218 \newglossarystyle{superragged3colborder}{%
```

Base it on the `glostylesuperragged3col` style:

```
8219 \setglossarystyle{superragged3col}%
```

Put the glossary in a `supertabular` environment with three columns and a horizontal line in the head and tail:

```
8220 \renewenvironment{theglossary}%  
8221 {\tablehead{\hline}\tabletail{\hline}%  
8222 \begin{supertabular}{|l|>{\raggedright}p{\glsdescwidth}|%  
8223 >{\raggedright}p{\glspagelistwidth}|}%  
8224 {\end{supertabular}}%  
8225 }
```

superragged3colheader The `superragged3colheader` style is like the `superragged3col` style but with a header row:

```
8226 \newglossarystyle{superragged3colheader}{%
```

Base it on the `glostylesuperragged3col` style:

```
8227 \setglossarystyle{superragged3col}%
```

Put the glossary in a `supertabular` environment with three columns, a header and no tail:

```
8228 \renewenvironment{theglossary}%  
8229 {\tablehead{\bfseries\entryname&\bfseries\descriptionname&  
8230 \bfseries\pagelistname\tabularnewline}\tabletail{}}%  
8231 \begin{supertabular}{|l>{\raggedright}p{\glsdescwidth}%  
8232 >{\raggedright}p{\glspagelistwidth}|}%  
8233 {\end{supertabular}}%  
8234 }
```

superragged3colheaderborder The `superragged3colheaderborder` style is like the `superragged3col` style but with a header and border:

```
8235 \newglossarystyle{superragged3colheaderborder}{%
```

Base it on the `glostylesuperragged3colborder` style:

```
8236 \setglossarystyle{superragged3colborder}%
```

Put the glossary in a `supertabular` environment with three columns, a header with horizontal lines and a horizontal line in the tail:

```
8237 \renewenvironment{theglossary}%  
8238 {\tablehead{\hline  
8239 \bfseries\entryname&\bfseries\descriptionname&  
8240 \bfseries\pagelistname\tabularnewline\hline}%
```

```

8241 \tabletail{\hline}%
8242 \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}|}%
8243 >{\raggedright}p{\glspagelistwidth}|}%
8244 {\end{supertabular}}%
8245 }

```

`altsuperragged4col` The `altsuperragged4col` glossary style is like `altsuper4col` style in the package but uses ragged right formatting in the description and page list columns.

```

8246 \newglossarystyle{altsuperragged4col}{%

```

Put the glossary in a `supertabular` environment with four columns and no head or tail:

```

8247 \renewenvironment{theglossary}%
8248 {\tablehead{}\tabletail{}}%
8249 \begin{supertabular}{|l>{\raggedright}p{\glsgdescwidth}l%
8250 >{\raggedright}p{\glspagelistwidth}|}%
8251 {\end{supertabular}}%

```

Do nothing at the start of the table:

```

8252 \renewcommand*{\glossaryheader}{}%

```

No group headings:

```

8253 \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:

```

8254 \renewcommand{\glossentry}[2]{%
8255 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
8256 \glossentrydesc{##1} &
8257 \glossentrysymbol{##1} & ##2\tabularnewline
8258 }%

```

Sub entries on a row with no name, the description in the second column, symbol in third column and page list in last column:

```

8259 \renewcommand{\subglossentry}[3]{%
8260 &
8261 \glssubentryitem{##2}%
8262 \glstarget{##2}{\strut}\glossentrydesc{##2} &
8263 \glossentrysymbol{##2} & ##3\tabularnewline
8264 }%

```

Blank row between groups:

```

8265 \renewcommand*{\glsgroupskip}{\ifglsgnোগroupskip\else & & \tabularnewline\fi}%
8266 }

```

`perragged4colheader` The `altsuperragged4colheader` style is like the `altsuperragged4col` style but with a header row.

```

8267 \newglossarystyle{altsuperragged4colheader}{%

```

Base it on the `glostylealtsuperragged4col` style:

```

8268 \setglossarystyle{altsuperragged4col}%

```

Put the glossary in a supertabular environment with four columns, a header and no tail:

```

8269 \renewenvironment{theglossary}%
8270   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
8271     \bfseries\symbolname &
8272     \bfseries\pagelistname\tabularnewline}\tabletail{}}%
8273   \begin{supertabular}{l>{\raggedright}p{\glsgdescwidth}l%
8274     >{\raggedright}p{\glspagelistwidth}}}%
8275   {\end{supertabular}}%
8276 }

```

`altperragged4colborder` The `altperragged4colborder` style is like the `altperragged4col` style but with a border.

```

8277 \newglossarystyle{altperragged4colborder}{%

```

Base it on the `glostylealtperragged4col` style:

```

8278 \setglossarystyle{altperragged4col}%

```

Put the glossary in a supertabular environment with four columns and a horizontal line in the head and tail:

```

8279 \renewenvironment{theglossary}%
8280   {\tablehead{\hline}\tabletail{\hline}%
8281   \begin{supertabular}%
8282     {\l|>{\raggedright}p{\glsgdescwidth}l|}%
8283     >{\raggedright}p{\glspagelistwidth}l}%
8284   {\end{supertabular}}%
8285 }

```

`altperragged4colheaderborder` The `altperragged4colheaderborder` style is like the `altperragged4col` style but with a header and border.

```

8286 \newglossarystyle{altperragged4colheaderborder}{%

```

Base it on the `glostylealtperragged4col` style:

```

8287 \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:

```

8288 \renewenvironment{theglossary}%
8289   {\tablehead{\hline
8290     \bfseries\entryname &
8291     \bfseries\descriptionname &
8292     \bfseries\symbolname &
8293     \bfseries\pagelistname\tabularnewline\hline}%
8294   \tabletail{\hline}%
8295   \begin{supertabular}%
8296     {\l|>{\raggedright}p{\glsgdescwidth}l|}%
8297     >{\raggedright}p{\glspagelistwidth}l}%
8298   {\end{supertabular}}%
8299 }

```


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.

8300 \ProvidesPackage{glossary-tree}[2014/08/27 v4.10 (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.

8301 \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.

8302 \newglossarystyle{index}{%

Set the paragraph indentation and skip and define \item to be the same as that used by theindex:

8303 \renewenvironment{theglossary}{%
8304 {\setlength{\parindent}{0pt}}%
8305 \setlength{\parskip}{0pt plus 0.3pt}}%
8306 \let\item\@idxitem}%
8307 {\par}}%

Do nothing at the start of the environment:

8308 \renewcommand*{\glossaryheader}{}%

No group headings:

8309 \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.

8310 \renewcommand*{\glossentry}[2]{%
8311 \item\glstentryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
8312 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
8313 \space \glossentrydesc{##1}\glspostdescription\space ##2%
8314 }%

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.

8315 \renewcommand{\subglossentry}[3]{%
8316 \ifcase##1\relax
8317 % level 0
8318 \item
8319 \or
8320 % level 1
8321 \subitem

```

8322     \glssubentryitem{##2}%
8323     \else
8324     % all other levels
8325     \subsubitem
8326     \fi
8327     \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}%
8328     \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
8329     \space\glossentrydesc{##2}\glspostdescription\space ##3%
8330 }%

```

Vertical gap between groups is the same as that used by indices:

```

8331 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}

```

indexgroup The indexgroup style is like the index style but has headings.

```

8332 \newglossarystyle{indexgroup}{%

```

Base it on the glostyleindex style:

```

8333 \setglossarystyle{index}%

```

Add a heading for each group. This puts the group's title in bold followed by a vertical gap.

```

8334 \renewcommand*{\glsgroupheading}[1]{%
8335 \item\glstreenamefmt{\glsgrouptitle{##1}}\indexspace}%
8336 }

```

indexhypergroup The indexhypergroup style is like the indexgroup style but has hyper navigation.

```

8337 \newglossarystyle{indexhypergroup}{%

```

Base it on the glostyleindex style:

```

8338 \setglossarystyle{index}%

```

Put navigation links to the groups at the start of the glossary:

```

8339 \renewcommand*{\glossaryheader}{%
8340 \item\glstreenamefmt{\glsnavigation}\indexspace}%

```

Add a heading for each group (with a target). The group's title is in bold followed by a vertical gap.

```

8341 \renewcommand*{\glsgroupheading}[1]{%
8342 \item\glstreenamefmt{\glsnavigation\hypertarget{##1}{\glsgrouptitle{##1}}}%
8343 \indexspace}%
8344 }

```

tree The tree glossary style is similar in style to the index style, but can have arbitrary levels.

```

8345 \newglossarystyle{tree}{%

```

Set the paragraph indentation and skip:

```

8346 \renewenvironment{theglossary}%
8347 {\setlength{\parindent}{0pt}%
8348 \setlength{\parskip}{0pt plus 0.3pt}}%
8349 {}%

```

Do nothing at the start of the theglossary environment:

```
8350 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
8351 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries: name in bold, followed by symbol in brackets (if it exists), the description and the page list:

```
8352 \renewcommand{\glossentry}[2]{%
8353   \hangindent0pt\relax
8354   \parindent0pt\relax
8355   \glsentryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
8356   \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
8357   \space\glossentrydesc{##1}\glspostdescription\space##2\par
8358 }%
```

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.

```
8359 \renewcommand{\subglossentry}[3]{%
8360   \hangindent##1\glstreeindent\relax
8361   \parindent##1\glstreeindent\relax
8362   \ifnum##1=1\relax
8363     \glssubentryitem{##2}%
8364   \fi
8365   \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}%
8366   \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
8367   \space\glossentrydesc{##2}\glspostdescription\space ##3\par
8368 }%
```

Vertical gap between groups is the same as that used by indices:

```
8369 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}}
```

treegroup Like the tree style but the glossary groups have headings.

```
8370 \newglossarystyle{treegroup}{%
```

Base it on the glostyletree style:

```
8371 \setglossarystyle{tree}%
```

Each group has a heading (in bold) followed by a vertical gap):

```
8372 \renewcommand{\glsgroupheading}[1]{\par
8373   \noindent\glstreenamefmt{\glsgrouptitle{##1}}\par\indexspace}%
8374 }
```

treehypergroup The treehypergroup style is like the treegroup style, but has a set of links to the groups at the start of the glossary.

```
8375 \newglossarystyle{treehypergroup}{%
```

Base it on the glostyletree style:

```
8376 \setglossarystyle{tree}%
```

Put navigation links to the groups at the start of the theglossary environment:

```
8377 \renewcommand*{\glossaryheader}{%
8378 \par\noindent\glstreenamefmt{\glsnavigation}\par\indexspace}%

```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
8379 \renewcommand*{\glsgroupheading}[1]{%
8380 \par\noindent
8381 \glstreenamefmt{\glsnahypertarget{##1}{\glsgrouptitle{##1}}}\par
8382 \indexspace}%
8383 }

```

`\glstreeindent` Length governing left indent for each level of the tree style.

```
8384 \newlength\glstreeindent
8385 \setlength{\glstreeindent}{10pt}

```

`treenoname` The `treenoname` glossary style is like the tree style, but doesn't print the name or symbol for sub-levels.

```
8386 \newglossarystyle{treenoname}{%

```

Set the paragraph indentation and skip:

```
8387 \renewenvironment{theglossary}%
8388 {\setlength{\parindent}{0pt}%
8389 \setlength{\parskip}{0pt plus 0.3pt}}%
8390 {}%

```

No header:

```
8391 \renewcommand*{\glossaryheader}{}%

```

No group headings:

```
8392 \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.

```
8393 \renewcommand{\glossentry}[2]{%
8394 \hangindent0pt\relax
8395 \parindent0pt\relax
8396 \glstryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
8397 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
8398 \space\glossentrydesc{##1}\glspostdescription\space##2\par
8399 }%

```

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.

```
8400 \renewcommand{\subglossentry}[3]{%
8401 \hangindent##1\glstreeindent\relax
8402 \parindent##1\glstreeindent\relax
8403 \ifnum##1=1\relax
8404 \glssubentryitem{##2}%
8405 \fi
8406 \glstarget{##2}{\strut}%
8407 \glossentrydesc{##2}\glspostdescription\space##3\par
8408 }%

```

Vertical gap between groups is the same as that used by indices:

```
8409 \renewcommand*{\glsgroupskip}{\ifglsgroupskip\else\indexspace\fi}%
8410 }
```

treenonamegroup Like the `treenoname` style but the glossary groups have headings.

```
8411 \newglossarystyle{treenonamegroup}{%
```

Base it on the `glostytreetreenoname` style:

```
8412 \setglossarystyle{treenoname}%
```

Give each group a heading:

```
8413 \renewcommand{\glsgroupheading}[1]{\par
```

```
8414 \noindent\glstreenamefmt{\glsgrouptitle{##1}}\par\indexspace}%
```

```
8415 }
```

treenonamehypergroup The `treenonamehypergroup` style is like the `treenonamegroup` style, but has a set of links to the groups at the start of the glossary.

```
8416 \newglossarystyle{treenonamehypergroup}{%
```

Base it on the `glostytreetreenoname` style:

```
8417 \setglossarystyle{treenoname}%
```

Put navigation links to the groups at the start of the `theglossary` environment:

```
8418 \renewcommand*{\glossaryheader}{%
```

```
8419 \par\noindent\glstreenamefmt{\glsgroupnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
8420 \renewcommand*{\glsgroupheading}[1]{%
```

```
8421 \par\noindent
```

```
8422 \glstreenamefmt{\glsgrouphypertarget{##1}{\glsgrouptitle{##1}}}\par
```

```
8423 \indexspace}%
```

```
8424 }
```

\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.

```
8425 \newcommand*{\glissetwidest}[2][0]{%
```

```
8426 \expandafter\def\csname @glswidestname\romannumeral#1\endcsname{%
```

```
8427 #2}%
```

```
8428 }
```

\@glswidestname Initialise `\@glswidestname`.

```
8429 \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`.

```
8430 \newglossarystyle{alttree}{%
```

Redefine theglossary environment.

```
8431 \renewenvironment{theglossary}%
8432   {\def\@gls@prevlevel{-1}%
8433    \mbox{}\par}%
8434   {\par}%
```

Set the header and group headers to nothing.

```
8435 \renewcommand*\glossaryheader{}%
8436 \renewcommand*\glsgroupheading[1]{}%
```

Redefine the way that the level 0 entries are displayed.

```
8437 \renewcommand{\glosseentry}[2]{%
8438   \ifnum\@gls@prevlevel=0\relax
8439   \else
```

Find out how big the indentation should be by measuring the widest entry.

```
8440     \settowidth{\glstreeindent}{\glstreenamfmt{\@glswidestname\space}}%
8441   \fi
```

Set the hangindent and paragraph indent.

```
8442     \hangindent\glstreeindent
8443     \parindent\glstreeindent
```

Put the name to the left of the paragraph block.

```
8444     \makebox[0pt][r]{\makebox[\glstreeindent][l]{%
8445       \glsentryitem{##1}\glstreenamfmt{\glstarget{##1}{\glosseentryname{##1}}}}}%
```

If the symbol is missing, ignore it, otherwise put it in brackets.

```
8446     \ifglshassymbol{##1}{(\glosseentrysymbol{##1})\space}{}%
```

Do the description followed by the description terminator and location list.

```
8447     \glosseentrydesc{##1}\glspostdescription \space ##2\par
```

Set the previous level to 0.

```
8448     \def\@gls@prevlevel{0}%
8449   }%
```

Redefine the way sub-entries are displayed.

```
8450 \renewcommand{\subglosseentry}[3]{%
```

Increment and display the sub-entry counter if this is a level 1 entry and the sub-entry counter is in use.

```
8451   \ifnum##1=1\relax
8452     \glssubentryitem{##2}%
8453   \fi
```

If the level hasn't changed, keep the same settings, otherwise adjust \glstreeindent accordingly.

```
8454   \ifnum\@gls@prevlevel=##1\relax
8455   \else
```

Compute the widest entry for this level, or for level 0 if not defined for this level.
Store in \gls@tmplen

```

8456     \@ifundefined{@glswidestname\romannumeral##1}{%
8457         \settowidth{\gls@tmplen}{\glstreenamefmt{\@glswidestname\space}}}%
8458         \settowidth{\gls@tmplen}{\glstreenamefmt{%
8459             \csname @glswidestname\romannumeral##1\endcsname\space}}}%

```

Determine if going up or down a level

```

8460     \ifnum\@gls@prevlevel<##1\relax

```

Depth has increased, so add the width of the widest entry to \glstreeindent.

```

8461         \setlength\glstreeindent\gls@tmplen
8462         \addtolength\glstreeindent\parindent
8463         \parindent\glstreeindent
8464     \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.

```

8465         \@ifundefined{@glswidestname\romannumeral\@gls@prevlevel}{%
8466             \settowidth{\glstreeindent}{\glstreenamefmt{%
8467                 \@glswidestname\space}}}%
8468             \settowidth{\glstreeindent}{\glstreenamefmt{%
8469                 \csname @glswidestname\romannumeral\@gls@prevlevel
8470                     \endcsname\space}}}%

```

Subtract this length from the previous level's paragraph indent and set to \glstreeindent.

```

8471         \addtolength\parindent{-\glstreeindent}%
8472         \setlength\glstreeindent\parindent
8473     \fi
8474 \fi

```

Set the hanging indentation.

```

8475     \hangindent\glstreeindent

```

Put the name to the left of the paragraph block

```

8476     \makebox[0pt][r]{\makebox[\gls@tmplen][l]{%
8477         \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}}}%

```

If the symbol is missing, ignore it, otherwise put it in brackets.

```

8478     \ifglshassymbol{##2}{(\glossentrysymbol{##2})\space}{}%

```

Do the description followed by the description terminator and location list.

```

8479     \glossentrydesc{##2}\glspostdescription\space ##3\par

```

Set the previous level macro to the current level.

```

8480     \def\@gls@prevlevel{##1}%
8481 }%

```

Vertical gap between groups is the same as that used by indices:

```

8482 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
8483 }

```

alttreegroup Like the `alttree` style but the glossary groups have headings.

```
8484 \newglossarystyle{alttreegroup}{%
      Base it on the glostylealttree style:
8485   \setglossarystyle{alttree}%
      Give each group a heading.
8486   \renewcommand{\glsgroupheading}[1]{\par
8487     \def\@gls@prevlevel{-1}%
8488     \hangindent0pt\relax
8489     \parindent0pt\relax
8490     \glstreenamefmt{\glsgetgrouptitle{##1}}\par\indexspace}%
8491 }
```

alttreehypergroup The `alttreehypergroup` style is like the `alttreegroup` style, but has a set of links to the groups at the start of the glossary.

```
8492 \newglossarystyle{alttreehypergroup}{%
      Base it on the glostylealttree style:
8493   \setglossarystyle{alttree}%
      Put the navigation links in the header
8494   \renewcommand*\glossaryheader{%
8495     \par
8496     \def\@gls@prevlevel{-1}%
8497     \hangindent0pt\relax
8498     \parindent0pt\relax
8499     \glstreenamefmt{\glsnavigation}\par\indexspace}%
      Put a hypertarget at the start of each group
8500   \renewcommand*\glsgroupheading[1]{%
8501     \par
8502     \def\@gls@prevlevel{-1}%
8503     \hangindent0pt\relax
8504     \parindent0pt\relax
8505     \glstreenamefmt{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
8506     \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.

```
8507 \NeedsTeXFormat{LaTeX2e}
8508 \ProvidesPackage{glossaries-compatible-207}[2011/04/02 v1.0 (NLCT)]
```

\GlsAddXdyAttribute Adds an attribute in old format.

```
8509 \ifglsxindy
8510   \renewcommand*\GlsAddXdyAttribute[1]{%
```



```

8511 \edef\@xdyattributes{\@xdyattributes ^^J \string"#1\string"}%
8512 \expandafter\toks@\expandafter{\@xdylocref}%
8513 \edef\@xdylocref{\the\toks@ ^^J%
8514 (markup-locref
8515 :open \string"\string~n\string\setentrycounter
8516 {\noexpand\glscounter}%
8517 \expandafter\string\csname#1\endcsname
8518 \expandafter\@gobble\string\{\string" ^^J
8519 :close \string"\expandafter\@gobble\string\}\string" ^^J
8520 :attr \string"#1\string")}}

```

Only has an effect before \writeist:

```

8521 \fi

```

\GlsAddXdyCounters

```

8522 \renewcommand*\GlsAddXdyCounters[1]{%
8523 \GlossariesWarning{\string\GlsAddXdyCounters\space not available
8524 in compatibility mode.}%
8525 }

```

Add predefined attributes

```

8526 \GlsAddXdyAttribute{glnumberformat}
8527 \GlsAddXdyAttribute{textrm}
8528 \GlsAddXdyAttribute{textsf}
8529 \GlsAddXdyAttribute{texttt}
8530 \GlsAddXdyAttribute{textbf}
8531 \GlsAddXdyAttribute{textmd}
8532 \GlsAddXdyAttribute{textit}
8533 \GlsAddXdyAttribute{textup}
8534 \GlsAddXdyAttribute{textsl}
8535 \GlsAddXdyAttribute{textsc}
8536 \GlsAddXdyAttribute{emph}
8537 \GlsAddXdyAttribute{glshypernumber}
8538 \GlsAddXdyAttribute{hyperrrm}
8539 \GlsAddXdyAttribute{hypersf}
8540 \GlsAddXdyAttribute{hypertt}
8541 \GlsAddXdyAttribute{hyperbf}
8542 \GlsAddXdyAttribute{hypermd}
8543 \GlsAddXdyAttribute{hyperit}
8544 \GlsAddXdyAttribute{hyperup}
8545 \GlsAddXdyAttribute{hypersl}
8546 \GlsAddXdyAttribute{hypersc}
8547 \GlsAddXdyAttribute{hyperemph}

```

\GlsAddXdyLocation Restore v2.07 definition:

```

8548 \ifglxindy
8549 \renewcommand*\GlsAddXdyLocation[2]{%
8550 \edef\@xdyuserlocationdefs{%
8551 \@xdyuserlocationdefs ^^J%

```

```

8552      (define-location-class \string"#1\string"^^J\space\space
8553      \space(#2))
8554    }%
8555    \edef\xdyuserlocationnames{%
8556      \xdyuserlocationnames^^J\space\space\space
8557      \string"#1\string"}%
8558  }
8559 \fi

```

\@do@wrglossary

```

8560 \renewcommand{\@do@wrglossary}[1]{%

```

Determine whether to use xindy or makeindex syntax

```

8561 \ifglsxindy

```

Need to determine if the formatting information starts with a (or) indicating a range.

```

8562 \expandafter\@glo@check@mkidxrangechar\@glsnumberformat\@nil
8563 \def\@glo@range{%
8564 \expandafter\if\@glo@prefix(\relax
8565 \def\@glo@range{:open-range}%
8566 \else
8567 \expandafter\if\@glo@prefix)\relax
8568 \def\@glo@range{:close-range}%
8569 \fi
8570 \fi

```

Get the location and escape any special characters

```

8571 \protected@edef\@glslocref{\theglentrycounter}%
8572 \@gls@checkmkidxchars\@glslocref

```

Write to the glossary file using xindy syntax.

```

8573 \glossary[\csname glo@#1@type\endcsname]{%
8574 (indexentry :key (\csname glo@#1@index\endcsname)
8575 :locref \string"\@glslocref\string" %
8576 :attr \string"\@glo@suffix\string" \@glo@range
8577 )
8578 }%
8579 \else

```

Convert the format information into the format required for makeindex

```

8580 \@set@glo@numformat\@glo@numfmt\@gls@counter\@glsnumberformat

```

Write to the glossary file using makeindex syntax.

```

8581 \glossary[\csname glo@#1@type\endcsname]{%
8582 \string\glossaryentry{\csname glo@#1@index\endcsname
8583 \@gls@encapchar\@glo@numfmt}\theglentrycounter}}%
8584 \fi
8585 }

```

\@set@glo@numformat Only had 3 arguments in v2.07

```

8586 \def\@set@glo@numformat#1#2#3{%
8587   \expandafter\@glo@check@mkidxrangechar#3\@nil
8588   \protected@edef#1{%
8589     \@glo@prefix setentrycounter[]{#2}%
8590     \expandafter\string\csname\@glo@suffix\endcsname
8591   }%
8592   \@gls@checkmkidxchars#1%
8593 }

```

\writeist Redefine \writeist back to the way it was in v2.07, but change \istfile to
\glswrite.

```

8594 \ifglxindy
8595   \def\writeist{%
8596     \openout\glswrite=\istfilename
8597     \write\glswrite{;; xindy style file created by the glossaries
8598       package in compatible-2.07 mode}%
8599     \write\glswrite{;; for document '\jobname' on
8600       \the\year-\the\month-\the\day}%
8601     \write\glswrite{^^J; required styles^^J}
8602     \@for\@xdystyle:=\@xdyrequiredstyles\do{%
8603       \ifx\@xdystyle\@empty
8604         \else
8605           \protected@write\glswrite{{(require
8606             \string"\@xdystyle.xdy\string")}}%
8607         \fi
8608       }%
8609     \write\glswrite{^^J%
8610       ; list of allowed attributes (number formats)^^J}%
8611     \write\glswrite{(define-attributes ((\@xdyattributes)))}%
8612     \write\glswrite{^^J; user defined alphabets^^J}%
8613     \write\glswrite{\@xdyuseralphabets}%
8614     \write\glswrite{^^J; location class definitions^^J}%
8615     \protected@edef\@gls@roman{\@roman{0}\string"
8616       \string"roman-numbers-lowercase\string" :sep \string"}}%
8617     \@onelevel@sanitize\@gls@roman
8618     \edef\@tmp{\string" \string"roman-numbers-lowercase\string"
8619       :sep \string"}%
8620     \@onelevel@sanitize\@tmp
8621     \ifx\@tmp\@gls@roman
8622       \write\glswrite{(define-location-class
8623         \string"roman-page-numbers\string"^^J\space\space\space
8624         (\string"roman-numbers-lowercase\string")
8625         :min-range-length \@glsminrange)}}%
8626     \else
8627       \write\glswrite{(define-location-class
8628         \string"roman-page-numbers\string"^^J\space\space\space
8629         (:sep "\@gls@roman")
8630         :min-range-length \@glsminrange)}}%
8631     \fi

```

```

8632 \write\glswrite{(define-location-class
8633   \string"Roman-page-numbers\string"^^J\space\space\space
8634   (\string"roman-numbers-uppercase\string")
8635   :min-range-length \@glsmminrange)}}%
8636 \write\glswrite{(define-location-class
8637   \string"arabic-page-numbers\string"^^J\space\space\space
8638   (\string"arabic-numbers\string")
8639   :min-range-length \@glsmminrange)}}%
8640 \write\glswrite{(define-location-class
8641   \string"alpha-page-numbers\string"^^J\space\space\space
8642   (\string"alpha\string")
8643   :min-range-length \@glsmminrange)}}%
8644 \write\glswrite{(define-location-class
8645   \string"Alpha-page-numbers\string"^^J\space\space\space
8646   (\string"ALPHA\string")
8647   :min-range-length \@glsmminrange)}}%
8648 \write\glswrite{(define-location-class
8649   \string"Appendix-page-numbers\string"^^J\space\space\space
8650   (\string"ALPHA\string"
8651   :sep \string"\@glAlphacompositor\string"
8652   \string"arabic-numbers\string")
8653   :min-range-length \@glsmminrange)}}%
8654 \write\glswrite{(define-location-class
8655   \string"arabic-section-numbers\string"^^J\space\space\space
8656   (\string"arabic-numbers\string"
8657   :sep \string"\glsc compositor\string"
8658   \string"arabic-numbers\string")
8659   :min-range-length \@glsmminrange)}}%
8660 \write\glswrite{^^J; user defined location classes}%
8661 \write\glswrite{@x dyuserlocationdefs}%
8662 \write\glswrite{^^J; define cross-reference class^^J}%
8663 \write\glswrite{(define-crossref-class \string"see\string"
8664   :unverified )}%
8665 \write\glswrite{(markup-crossref-list
8666   :class \string"see\string"^^J\space\space\space
8667   :open \string"\string\glseeformat\string"
8668   :close \string"{}\string")}%
8669 \write\glswrite{^^J; define the order of the location classes}%
8670 \write\glswrite{(define-location-class-order
8671   (\@x dylocationclassorder))}%
8672 \write\glswrite{^^J; define the glossary markup^^J}%
8673 \write\glswrite{(markup-index^^J\space\space\space
8674   :open \string"\string
8675   \glossarysection[\string\glossarytoctitle]{\string
8676   \glossarytitle}\string\glossarypreamble\string~n\string\begin
8677   {theglossary}\string\glossaryheader\string~n\string" ^^J\space
8678   \space\space:close \string"\expandafter\@gobble
8679   \string\%\string~n\string
8680   \end{theglossary}\string\glossarypostamble

```

```

8681         \string~n\string" ^^J\space\space\space
8682     :tree)}}%
8683 \write\glswrite{(markup-letter-group-list
8684     :sep \string"\string\glsgroupskip\string~n\string"))}%
8685 \write\glswrite{(markup-indexentry
8686     :open \string"\string\relax \string\glsresetentrylist
8687         \string~n\string"))}%
8688 \write\glswrite{(markup-locclass-list :open
8689     \string"\glsopenbrace\string\glossaryentrynumbers
8690     \glsopenbrace\string\relax\space \string"^^J\space\space\space
8691     :sep \string", \string"
8692     :close \string"\glsclosebrace\glsclosebrace\string"))}%
8693 \write\glswrite{(markup-locref-list
8694     :sep \string"\string\delimN\space\string"))}%
8695 \write\glswrite{(markup-range
8696     :sep \string"\string\delimR\space\string"))}%
8697 \@onelevel@sanitize\gls@suffixF
8698 \@onelevel@sanitize\gls@suffixFF
8699 \ifx\gls@suffixF\@empty
8700 \else
8701     \write\glswrite{(markup-range
8702         :close "\gls@suffixF" :length 1 :ignore-end)}}%
8703 \fi
8704 \ifx\gls@suffixFF\@empty
8705 \else
8706     \write\glswrite{(markup-range
8707         :close "\gls@suffixFF" :length 2 :ignore-end)}}%
8708 \fi
8709 \write\glswrite{^^J; define format to use for locations^^J}%
8710 \write\glswrite{\@xdylocref}%
8711 \write\glswrite{^^J; define letter group list format^^J}%
8712 \write\glswrite{(markup-letter-group-list
8713     :sep \string"\string\glsgroupskip\string~n\string"))}%
8714 \write\glswrite{^^J; letter group headings^^J}%
8715 \write\glswrite{(markup-letter-group
8716     :open-head \string"\string\glsgroupheading
8717     \glsopenbrace\string"^^J\space\space\space
8718     :close-head \string"\glsclosebrace\string"))}%
8719 \write\glswrite{^^J; additional letter groups^^J}%
8720 \write\glswrite{\@xdylettergroups}%
8721 \write\glswrite{^^J; additional sort rules^^J}%
8722 \write\glswrite{\@dysortrules}%
8723 \noist}
8724 \else
8725     \edef\@gls@actualchar{\string?}
8726     \edef\@gls@encapchar{\string|}
8727     \edef\@gls@levelchar{\string!}
8728     \edef\@gls@quotechar{\string"}
8729     \def\writeist{\relax

```

```

8730 \openout\glswrite=\istfilename
8731 \write\glswrite{\expandafter\@gobble\string\% makeindex style file
8732   created by the glossaries package}
8733 \write\glswrite{\expandafter\@gobble\string\% for document
8734   '\jobname' on \the\year-\the\month-\the\day}
8735 \write\glswrite{actual '\@gls@actualchar'}
8736 \write\glswrite{encap '\@gls@encapchar'}
8737 \write\glswrite{level '\@gls@levelchar'}
8738 \write\glswrite{quote '\@gls@quotechar'}
8739 \write\glswrite{keyword \string"\string\glossaryentry\string"}
8740 \write\glswrite{preamble \string"\string\glossarysection[\string
8741   \glossarytoctitle]{\string\glossarytitle}\string
8742   \glossarypreamble\string\n\string\begin{theglossary}\string
8743   \glossaryheader\string\n\string"}
8744 \write\glswrite{postamble \string"\string\%\string\n\string
8745   \end{theglossary}\string\glossarypostamble\string\n
8746   \string"}
8747 \write\glswrite{group_skip \string"\string\glsgroupskip\string\n
8748   \string"}
8749 \write\glswrite{item_0 \string"\string\%\string\n\string"}
8750 \write\glswrite{item_1 \string"\string\%\string\n\string"}
8751 \write\glswrite{item_2 \string"\string\%\string\n\string"}
8752 \write\glswrite{item_01 \string"\string\%\string\n\string"}
8753 \write\glswrite{item_x1
8754   \string"\string\relax \string\glsresetentrylist\string\n
8755   \string"}
8756 \write\glswrite{item_12 \string"\string\%\string\n\string"}
8757 \write\glswrite{item_x2
8758   \string"\string\relax \string\glsresetentrylist\string\n
8759   \string"}
8760 \write\glswrite{delim_0 \string"\string\{\string
8761   \glossaryentrynumbers\string\{\string\relax \string"}
8762 \write\glswrite{delim_1 \string"\string\{\string
8763   \glossaryentrynumbers\string\{\string\relax \string"}
8764 \write\glswrite{delim_2 \string"\string\{\string
8765   \glossaryentrynumbers\string\{\string\relax \string"}
8766 \write\glswrite{delim_t \string"\string\}\string\}\string"}
8767 \write\glswrite{delim_n \string"\string\delimN \string"}
8768 \write\glswrite{delim_r \string"\string\delimR \string"}
8769 \write\glswrite{headings_flag 1}
8770 \write\glswrite{heading_prefix
8771   \string"\string\glsgroupheading\string\{\string"}
8772 \write\glswrite{heading_suffix
8773   \string"\string\}\string\relax
8774   \string\glsresetentrylist \string"}
8775 \write\glswrite{symhead_positive \string"glssymbols\string"}
8776 \write\glswrite{numhead_positive \string"glslnumbers\string"}
8777 \write\glswrite{page_compositor \string"glscpositor\string"}
8778 \@gls@escbsdq\gls@suffixF

```

```

8779 \@gls@escbsdq\gls@suffixFF
8780 \ifx\gls@suffixF\@empty
8781 \else
8782 \write\glswrite{suffix_2p \string"\gls@suffixF\string"}
8783 \fi
8784 \ifx\gls@suffixFF\@empty
8785 \else
8786 \write\glswrite{suffix_3p \string"\gls@suffixFF\string"}
8787 \fi
8788 \noist
8789 }
8790 \fi

```

\noist

```

8791 \renewcommand*{\noist}{\let\writeist\relax}

```

Compatibility macros.

```

8792 \NeedsTeXFormat{LaTeX2e}
8793 \ProvidesPackage{glossaries-compatible-307}[2013/11/14 v4.0 (NLCT)]

```

Compatibility macros for predefined glossary styles:

compatglossarystyle Defines a compatibility glossary style.

```

8794 \newcommand{\compatglossarystyle}[2]{%
8795 \ifcsundef{@glscompstyle@#1}%
8796 {%
8797 \csdef{@glscompstyle@#1}{#2}%
8798 }%
8799 {%
8800 \PackageError{glossaries}{Glossary compatibility style ‘#1’ is already defined}{}%
8801 }%
8802 }

```

Backward compatible inline style.

```

8803 \compatglossarystyle{inline}{%
8804 \renewcommand{\glossaryentryfield}[5]{%
8805 \glsinlinedopostchild
8806 \gls@inlinesep
8807 \def\glo@desc{##3}%
8808 \def\@no@post@desc{\nopostdesc}%
8809 \glsentryitem{##1}\glsinlinenameformat{##1}{##2}%
8810 \ifx\glo@desc\@no@post@desc
8811 \glsinlineemptydescformat{##4}{##5}%
8812 \else
8813 \ifstrempy{##3}%
8814 {\glsinlineemptydescformat{##4}{##5}}%
8815 {\glsinlinedescformat{##3}{##4}{##5}}%
8816 \fi
8817 \ifglshaschildren{##1}%
8818 {%

```

```

8819      \glsresetsubentrycounter
8820      \glsinlineparentchildseparator
8821      \def\gls@inlinesubsep{}%
8822      \def\gls@inlinepostchild{\glsinlinepostchild}%
8823      }%
8824      {}%
8825      \def\gls@inlinesep{\glsinlineseparator}%
8826      }%

```

Sub-entries display description:

```

8827      \renewcommand{\glossarysubentryfield}[6]{%
8828          \gls@inlinesubsep%
8829          \glsinlinesubnameformat{##2}{##3}%
8830          \glssubentryitem{##2}\glsinlinesubdescformat{##4}{##5}{##6}%
8831          \def\gls@inlinesubsep{\glsinlinesubseparator}%
8832      }%
8833 }

```

Backward compatible list style.

```

8834 \compatglossarystyle{list}{%
8835     \renewcommand*{\glossaryentryfield}[5]{%
8836         \item[\glsentryitem{##1}\glstarget{##1}{##2}]
8837         ##3\glspostdescription\space ##5}%

```

Sub-entries continue on the same line:

```

8838     \renewcommand*{\glossarysubentryfield}[6]{%
8839         \glssubentryitem{##2}%
8840         \glstarget{##2}{\strut}##4\glspostdescription\space ##6.}%
8841 }

```

Backward compatible listgroup style.

```

8842 \compatglossarystyle{listgroup}{%
8843     \csuse{@glscompstyle@list}%
8844 }%

```

Backward compatible listhypergroup style.

```

8845 \compatglossarystyle{listhypergroup}{%
8846     \csuse{@glscompstyle@list}%
8847 }%

```

Backward compatible altlist style.

```

8848 \compatglossarystyle{altlist}{%
8849     \renewcommand*{\glossaryentryfield}[5]{%
8850         \item[\glsentryitem{##1}\glstarget{##1}{##2}]%
8851         \mbox{}\par\nobreak\@afterheading
8852         ##3\glspostdescription\space ##5}%
8853     \renewcommand{\glossarysubentryfield}[6]{%
8854         \par
8855         \glssubentryitem{##2}%
8856         \glstarget{##2}{\strut}##4\glspostdescription\space ##6}%
8857 }%

```


Backward compatible altlistgroup style.

```
8858 \compatglossarystyle{altlistgroup}{%  
8859 \csuse{@glscompstyle@altlist}%  
8860 }%
```

Backward compatible altlisthypergroup style.

```
8861 \compatglossarystyle{altlisthypergroup}{%  
8862 \csuse{@glscompstyle@altlist}%  
8863 }%
```

Backward compatible listdotted style.

```
8864 \compatglossarystyle{listdotted}{%  
8865 \renewcommand*{\glossaryentryfield}[5]{%  
8866 \item[]\makebox[\glslistdottedwidth][l]{%  
8867 \glentryitem{##1}\glstarget{##1}{##2}%  
8868 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##3}%  
8869 \renewcommand*{\glossarysubentryfield}[6]{%  
8870 \item[]\makebox[\glslistdottedwidth][l]{%  
8871 \glssubentryitem{##2}%  
8872 \glstarget{##2}{##3}%  
8873 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##4}%  
8874 }%
```

Backward compatible sublistdotted style.

```
8875 \compatglossarystyle{sublistdotted}{%  
8876 \csuse{@glscompstyle@listdotted}%  
8877 \renewcommand*{\glossaryentryfield}[5]{%  
8878 \item[\glentryitem{##1}\glstarget{##1}{##2}]}%  
8879 }%
```

Backward compatible long style.

```
8880 \compatglossarystyle{long}{%  
8881 \renewcommand*{\glossaryentryfield}[5]{%  
8882 \glentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\}%  
8883 \renewcommand*{\glossarysubentryfield}[6]{%  
8884 &  
8885 \glssubentryitem{##2}%  
8886 \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\}%  
8887 }%
```

Backward compatible longborder style.

```
8888 \compatglossarystyle{longborder}{%  
8889 \csuse{@glscompstyle@long}%  
8890 }%
```

Backward compatible longheader style.

```
8891 \compatglossarystyle{longheader}{%  
8892 \csuse{@glscompstyle@long}%  
8893 }%
```

Backward compatible longheaderborder style.

```
8894 \compatglossarystyle{longheaderborder}{%
```

```

8895 \csuse{@glscompstyle@long}%
8896 }%

```

Backward compatible long3col style.

```

8897 \compatglossarystyle{long3col}{%
8898   \renewcommand*{\glossaryentryfield}[5]{%
8899     \glsentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\\}%
8900   \renewcommand*{\glossarysubentryfield}[6]{%
8901     &
8902     \glssubentryitem{##2}%
8903     \glstarget{##2}{\strut}##4 & ##6\\}%
8904 }%

```

Backward compatible long3colborder style.

```

8905 \compatglossarystyle{long3colborder}{%
8906   \csuse{@glscompstyle@long3col}%
8907 }%

```

Backward compatible long3colheader style.

```

8908 \compatglossarystyle{long3colheader}{%
8909   \csuse{@glscompstyle@long3col}%
8910 }%

```

Backward compatible long3colheaderborder style.

```

8911 \compatglossarystyle{long3colheaderborder}{%
8912   \csuse{@glscompstyle@long3col}%
8913 }%

```

Backward compatible long4col style.

```

8914 \compatglossarystyle{long4col}{%
8915   \renewcommand*{\glossaryentryfield}[5]{%
8916     \glsentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\\}%
8917   \renewcommand*{\glossarysubentryfield}[6]{%
8918     &
8919     \glssubentryitem{##2}%
8920     \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%
8921 }%

```

Backward compatible long4colheader style.

```

8922 \compatglossarystyle{long4colheader}{%
8923   \csuse{@glscompstyle@long4col}%
8924 }%

```

Backward compatible long4colborder style.

```

8925 \compatglossarystyle{long4colborder}{%
8926   \csuse{@glscompstyle@long4col}%
8927 }%

```

Backward compatible long4colheaderborder style.

```

8928 \compatglossarystyle{long4colheaderborder}{%
8929   \csuse{@glscompstyle@long4col}%
8930 }%

```

Backward compatible altlong4col style.

```
8931 \compatglossarystyle{altlong4col}{%  
8932 \csuse{@glscompstyle@long4col}%  
8933 }%
```

Backward compatible altlong4colheader style.

```
8934 \compatglossarystyle{altlong4colheader}{%  
8935 \csuse{@glscompstyle@long4col}%  
8936 }%
```

Backward compatible altlong4colborder style.

```
8937 \compatglossarystyle{altlong4colborder}{%  
8938 \csuse{@glscompstyle@long4col}%  
8939 }%
```

Backward compatible altlong4colheaderborder style.

```
8940 \compatglossarystyle{altlong4colheaderborder}{%  
8941 \csuse{@glscompstyle@long4col}%  
8942 }%
```

Backward compatible long style.

```
8943 \compatglossarystyle{longragged}{%  
8944 \renewcommand*{\glossaryentryfield}[5]{%  
8945 \glstryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%  
8946 \tabularnewline}%  
8947 \renewcommand*{\glossarysubentryfield}[6]{%  
8948 &  
8949 \glssubentryitem{##2}%  
8950 \glstarget{##2}{\strut}##4\glspostdescription\space ##6%  
8951 \tabularnewline}%  
8952 }%
```

Backward compatible longraggedborder style.

```
8953 \compatglossarystyle{longraggedborder}{%  
8954 \csuse{@glscompstyle@longragged}%  
8955 }%
```

Backward compatible longraggedheader style.

```
8956 \compatglossarystyle{longraggedheader}{%  
8957 \csuse{@glscompstyle@longragged}%  
8958 }%
```

Backward compatible longraggedheaderborder style.

```
8959 \compatglossarystyle{longraggedheaderborder}{%  
8960 \csuse{@glscompstyle@longragged}%  
8961 }%
```

Backward compatible longragged3col style.

```
8962 \compatglossarystyle{longragged3col}{%  
8963 \renewcommand*{\glossaryentryfield}[5]{%  
8964 \glstryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\tabularnewline}%  
8965 \renewcommand*{\glossarysubentryfield}[6]{%
```

```

8966      &
8967      \glssubentryitem{##2}%
8968      \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%
8969 }%

```

Backward compatible longragged3colborder style.

```

8970 \compatglossarystyle{longragged3colborder}{%
8971   \csuse{@glscmpstyle@longragged3col}%
8972 }%

```

Backward compatible longragged3colheader style.

```

8973 \compatglossarystyle{longragged3colheader}{%
8974   \csuse{@glscmpstyle@longragged3col}%
8975 }%

```

Backward compatible longragged3colheaderborder style.

```

8976 \compatglossarystyle{longragged3colheaderborder}{%
8977   \csuse{@glscmpstyle@longragged3col}%
8978 }%

```

Backward compatible altlongragged4col style.

```

8979 \compatglossarystyle{altlongragged4col}{%
8980   \renewcommand*{\glossaryentryfield}[5]{%
8981     \glssentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\tabularnewline}%
8982   \renewcommand*{\glossarysubentryfield}[6]{%
8983     &
8984     \glssubentryitem{##2}%
8985     \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%
8986 }%

```

Backward compatible altlongragged4colheader style.

```

8987 \compatglossarystyle{altlongragged4colheader}{%
8988   \csuse{@glscmpstyle@altlong4col}%
8989 }%

```

Backward compatible altlongragged4colborder style.

```

8990 \compatglossarystyle{altlongragged4colborder}{%
8991   \csuse{@glscmpstyle@altlong4col}%
8992 }%

```

Backward compatible altlongragged4colheaderborder style.

```

8993 \compatglossarystyle{altlongragged4colheaderborder}{%
8994   \csuse{@glscmpstyle@altlong4col}%
8995 }%

```

Backward compatible index style.

```

8996 \compatglossarystyle{index}{%
8997   \renewcommand*{\glossaryentryfield}[5]{%
8998     \item\glssentryitem{##1}\textbf{\glstarget{##1}{##2}}%
8999     \ifx\relax##4\relax
9000     \else
9001     \space{##4}%

```

```

9002     \fi
9003     \space ##3\glspostdescription \space ##5}%
9004 \renewcommand*{\glossarysubentryfield}[6]{%
9005     \ifcase##1\relax
9006     % level 0
9007     \item
9008     \or
9009     % level 1
9010     \subitem
9011     \glssubentryitem{##2}%
9012     \else
9013     % all other levels
9014     \subsubitem
9015     \fi
9016     \textbf{\glstarget{##2}{##3}}%
9017     \ifx\relax##5\relax
9018     \else
9019     \space(##5)%
9020     \fi
9021     \space##4\glspostdescription\space ##6}%
9022 }%

```

Backward compatible indexgroup style.

```

9023 \compatglossarystyle{indexgroup}{%
9024 \csuse{@glscmpstyle@index}%
9025 }%

```

Backward compatible indexhypergroup style.

```

9026 \compatglossarystyle{indexhypergroup}{%
9027 \csuse{@glscmpstyle@index}%
9028 }%

```

Backward compatible tree style.

```

9029 \compatglossarystyle{tree}{%
9030 \renewcommand{\glossaryentryfield}[5]{%
9031     \hangindent0pt\relax
9032     \parindent0pt\relax
9033     \glstentryitem{##1}\textbf{\glstarget{##1}{##2}}%
9034     \ifx\relax##4\relax
9035     \else
9036     \space(##4)%
9037     \fi
9038     \space ##3\glspostdescription \space ##5\par}%
9039 \renewcommand{\glossarysubentryfield}[6]{%
9040     \hangindent##1\glstreeindent\relax
9041     \parindent##1\glstreeindent\relax
9042     \ifnum##1=1\relax
9043     \glssubentryitem{##2}%
9044     \fi
9045     \textbf{\glstarget{##2}{##3}}%
9046     \ifx\relax##5\relax

```

```

9047 \else
9048 \space(##5)%
9049 \fi
9050 \space##4\glspostdescription\space ##6\par}%
9051 }%

Backward compatible treegroup style.
9052 \compatglossarystyle{treegroup}{%
9053 \csuse{@glscmpstyle@tree}%
9054 }%

Backward compatible treehypergroup style.
9055 \compatglossarystyle{treehypergroup}{%
9056 \csuse{@glscmpstyle@tree}%
9057 }%

Backward compatible treenoname style.
9058 \compatglossarystyle{treenoname}{%
9059 \renewcommand{\glossaryentryfield}[5]{%
9060 \hangindent0pt\relax
9061 \parindent0pt\relax
9062 \glstentryitem{##1}\textbf{\glstarget{##1}{##2}}%
9063 \ifx\relax##4\relax
9064 \else
9065 \space(##4)%
9066 \fi
9067 \space ##3\glspostdescription \space ##5\par}%
9068 \renewcommand{\glossarysubentryfield}[6]{%
9069 \hangindent##1\glstreeindent\relax
9070 \parindent##1\glstreeindent\relax
9071 \ifnum##1=1\relax
9072 \glssubentryitem{##2}%
9073 \fi
9074 \glstarget{##2}{\strut}%
9075 ##4\glspostdescription\space ##6\par}%
9076 }%

Backward compatible treenonamegroup style.
9077 \compatglossarystyle{treenonamegroup}{%
9078 \csuse{@glscmpstyle@treenoname}%
9079 }%

Backward compatible treenonamehypergroup style.
9080 \compatglossarystyle{treenonamehypergroup}{%
9081 \csuse{@glscmpstyle@treenoname}%
9082 }%

Backward compatible alttree style.
9083 \compatglossarystyle{alttree}{%
9084 \renewcommand{\glossaryentryfield}[5]{%
9085 \ifnum \@gls@prevlevel=0\relax
9086 \else

```

```

9087     \settowidth{\glstreeindent}{\textbf{\@glswidestname\space}}%
9088     \hangindent\glstreeindent
9089     \parindent\glstreeindent
9090     \fi
9091     \makebox[0pt][r]{\makebox[\glstreeindent][l]{%
9092       \glstentryitem{##1}\textbf{\glstarget{##1}{##2}}}%
9093     \ifx\relax##4\relax
9094     \else
9095       (##4)\space
9096     \fi
9097     ##3\glspostdescription \space ##5\par
9098     \def\@gls@prevlevel{0}%
9099   }%
9100   \renewcommand{\glossarysubentryfield}[6]{%
9101     \ifnum##1=1\relax
9102       \glssubentryitem{##2}%
9103     \fi
9104     \ifnum\@gls@prevlevel=##1\relax
9105     \else
9106       \@ifundefined{\@glswidestname\romannumeral##1}{%
9107         \settowidth{\gls@tmplen}{\textbf{\@glswidestname\space}}{%
9108         \settowidth{\gls@tmplen}{\textbf{%
9109           \csname \@glswidestname\romannumeral##1\endcsname\space}}}%
9110       \ifnum\@gls@prevlevel<##1\relax
9111         \setlength\glstreeindent\gls@tmplen
9112         \addtolength\glstreeindent\parindent
9113         \parindent\glstreeindent
9114       \else
9115         \@ifundefined{\@glswidestname\romannumeral\@gls@prevlevel}{%
9116         \settowidth{\glstreeindent}{\textbf{%
9117           \@glswidestname\space}}{%
9118         \settowidth{\glstreeindent}{\textbf{%
9119           \csname \@glswidestname\romannumeral\@gls@prevlevel
9120             \endcsname\space}}}%
9121         \addtolength\parindent{-\glstreeindent}%
9122         \setlength\glstreeindent\parindent
9123       \fi
9124     \fi
9125     \hangindent\glstreeindent
9126     \makebox[0pt][r]{\makebox[\gls@tmplen][l]{%
9127       \textbf{\glstarget{##2}{##3}}}%
9128     \ifx##5\relax\relax
9129     \else
9130       (##5)\space
9131     \fi
9132     ##4\glspostdescription\space ##6\par
9133     \def\@gls@prevlevel{##1}%
9134   }%
9135 }%

```

Backward compatible alttreegroup style.

```
9136 \compatglossarystyle{alttreegroup}{%  
9137 \csuse{@glscompstyle@alttree}%  
9138 }%
```

Backward compatible alttreehypergroup style.

```
9139 \compatglossarystyle{alttreehypergroup}{%  
9140 \csuse{@glscompstyle@alttree}%  
9141 }%
```

Backward compatible mcolindex style.

```
9142 \compatglossarystyle{mcolindex}{%  
9143 \csuse{@glscompstyle@index}%  
9144 }%
```

Backward compatible mcolindexgroup style.

```
9145 \compatglossarystyle{mcolindexgroup}{%  
9146 \csuse{@glscompstyle@index}%  
9147 }%
```

Backward compatible mcolindexhypergroup style.

```
9148 \compatglossarystyle{mcolindexhypergroup}{%  
9149 \csuse{@glscompstyle@index}%  
9150 }%
```

Backward compatible mcoltree style.

```
9151 \compatglossarystyle{mcoltree}{%  
9152 \csuse{@glscompstyle@tree}%  
9153 }%
```

Backward compatible mcoltreegroup style.

```
9154 \compatglossarystyle{mcolindextreegroup}{%  
9155 \csuse{@glscompstyle@tree}%  
9156 }%
```

Backward compatible mcoltreehypergroup style.

```
9157 \compatglossarystyle{mcolindextreehypergroup}{%  
9158 \csuse{@glscompstyle@tree}%  
9159 }%
```

Backward compatible mcoltreenoname style.

```
9160 \compatglossarystyle{mcoltreenoname}{%  
9161 \csuse{@glscompstyle@tree}%  
9162 }%
```

Backward compatible mcoltreenonamegroup style.

```
9163 \compatglossarystyle{mcoltreenonamegroup}{%  
9164 \csuse{@glscompstyle@tree}%  
9165 }%
```

Backward compatible mcoltreenonamehypergroup style.

```
9166 \compatglossarystyle{mcoltreenonamehypergroup}{%  
9167 \csuse{@glscompstyle@tree}%  
9168 }%
```


Backward compatible mcolalmtree style.

```
9169 \compatglossarystyle{mcolalmtree}{%  
9170 \csuse{@glscmpstyle@almtree}%  
9171 }%
```

Backward compatible mcolalmtreegroup style.

```
9172 \compatglossarystyle{mcolalmtreegroup}{%  
9173 \csuse{@glscmpstyle@almtree}%  
9174 }%
```

Backward compatible mcolalmtreehypergroup style.

```
9175 \compatglossarystyle{mcolalmtreehypergroup}{%  
9176 \csuse{@glscmpstyle@almtree}%  
9177 }%
```

Backward compatible superragged style.

```
9178 \compatglossarystyle{superragged}{%  
9179 \renewcommand*{\glossaryentryfield}[5]{%  
9180 \glstentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%  
9181 \tabularnewline}%  
9182 \renewcommand*{\glossarysubentryfield}[6]{%  
9183 &  
9184 \glssubentryitem{##2}%  
9185 \glstarget{##2}{\strut}##4\glspostdescription\space ##6%  
9186 \tabularnewline}%  
9187 }%
```

Backward compatible superraggedborder style.

```
9188 \compatglossarystyle{superraggedborder}{%  
9189 \csuse{@glscmpstyle@superragged}%  
9190 }%
```

Backward compatible superraggedheader style.

```
9191 \compatglossarystyle{superraggedheader}{%  
9192 \csuse{@glscmpstyle@superragged}%  
9193 }%
```

Backward compatible superraggedheaderborder style.

```
9194 \compatglossarystyle{superraggedheaderborder}{%  
9195 \csuse{@glscmpstyle@superragged}%  
9196 }%
```

Backward compatible superragged3col style.

```
9197 \compatglossarystyle{superragged3col}{%  
9198 \renewcommand*{\glossaryentryfield}[5]{%  
9199 \glstentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\tabularnewline}%  
9200 \renewcommand*{\glossarysubentryfield}[6]{%  
9201 &  
9202 \glssubentryitem{##2}%  
9203 \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%  
9204 }%
```

Backward compatible superragged3colborder style.

```
9205 \compatglossarystyle{superragged3colborder}{%
9206 \csuse{@glscmpstyle@superragged3col}%
9207 }%
```

Backward compatible superragged3colheader style.

```
9208 \compatglossarystyle{superragged3colheader}{%
9209 \csuse{@glscmpstyle@superragged3col}%
9210 }%
```

Backward compatible superragged3colheaderborder style.

```
9211 \compatglossarystyle{superragged3colheaderborder}{%
9212 \csuse{@glscmpstyle@superragged3col}%
9213 }%
```

Backward compatible altsuperragged4col style.

```
9214 \compatglossarystyle{altsuperragged4col}{%
9215 \renewcommand*{\glossaryentryfield}[5]{%
9216 \glentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\tabularnewline}%
9217 \renewcommand*{\glossarysubentryfield}[6]{%
9218 &
9219 \glssubentryitem{##2}%
9220 \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%
9221 }%
```

Backward compatible altsuperragged4colheader style.

```
9222 \compatglossarystyle{altsuperragged4colheader}{%
9223 \csuse{@glscmpstyle@altsuperragged4col}%
9224 }%
```

Backward compatible altsuperragged4colborder style.

```
9225 \compatglossarystyle{altsuperragged4colborder}{%
9226 \csuse{@glscmpstyle@altsuperragged4col}%
9227 }%
```

Backward compatible altsuperragged4colheaderborder style.

```
9228 \compatglossarystyle{altsuperragged4colheaderborder}{%
9229 \csuse{@glscmpstyle@altsuperragged4col}%
9230 }%
```

Backward compatible super style.

```
9231 \compatglossarystyle{super}{%
9232 \renewcommand*{\glossaryentryfield}[5]{%
9233 \glentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\}%
9234 \renewcommand*{\glossarysubentryfield}[6]{%
9235 &
9236 \glssubentryitem{##2}%
9237 \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\}%
9238 }%
```

Backward compatible superborder style.

```
9239 \compatglossarystyle{superborder}{%

```

```

9240 \csuse{@glscompstyle@super}%
9241 }%

```

Backward compatible superheader style.

```

9242 \compatglossarystyle{superheader}{%
9243 \csuse{@glscompstyle@super}%
9244 }%

```

Backward compatible superheaderborder style.

```

9245 \compatglossarystyle{superheaderborder}{%
9246 \csuse{@glscompstyle@super}%
9247 }%

```

Backward compatible super3col style.

```

9248 \compatglossarystyle{super3col}{%
9249 \renewcommand*{\glossaryentryfield}[5]{%
9250 \glstentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\\}%
9251 \renewcommand*{\glossarysubentryfield}[6]{%
9252 &
9253 \glssubentryitem{##2}%
9254 \glstarget{##2}{\strut}##4 & ##6\\}%
9255 }%

```

Backward compatible super3colborder style.

```

9256 \compatglossarystyle{super3colborder}{%
9257 \csuse{@glscompstyle@super3col}%
9258 }%

```

Backward compatible super3colheader style.

```

9259 \compatglossarystyle{super3colheader}{%
9260 \csuse{@glscompstyle@super3col}%
9261 }%

```

Backward compatible super3colheaderborder style.

```

9262 \compatglossarystyle{super3colheaderborder}{%
9263 \csuse{@glscompstyle@super3col}%
9264 }%

```

Backward compatible super4col style.

```

9265 \compatglossarystyle{super4col}{%
9266 \renewcommand*{\glossaryentryfield}[5]{%
9267 \glstentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\\}%
9268 \renewcommand*{\glossarysubentryfield}[6]{%
9269 &
9270 \glssubentryitem{##2}%
9271 \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%
9272 }%

```

Backward compatible super4colheader style.

```

9273 \compatglossarystyle{super4colheader}{%
9274 \csuse{@glscompstyle@super4col}%
9275 }%

```

Backward compatible super4colborder style.

```
9276 \compatglossarystyle{super4colborder}{%
9277 \csuse{@glscmpstyle@super4col}%
9278 }%
```

Backward compatible super4colheaderborder style.

```
9279 \compatglossarystyle{super4colheaderborder}{%
9280 \csuse{@glscmpstyle@super4col}%
9281 }%
```

Backward compatible altsuper4col style.

```
9282 \compatglossarystyle{altsuper4col}{%
9283 \csuse{@glscmpstyle@super4col}%
9284 }%
```

Backward compatible altsuper4colheader style.

```
9285 \compatglossarystyle{altsuper4colheader}{%
9286 \csuse{@glscmpstyle@super4col}%
9287 }%
```

Backward compatible altsuper4colborder style.

```
9288 \compatglossarystyle{altsuper4colborder}{%
9289 \csuse{@glscmpstyle@super4col}%
9290 }%
```

Backward compatible altsuper4colheaderborder style.

```
9291 \compatglossarystyle{altsuper4colheaderborder}{%
9292 \csuse{@glscmpstyle@super4col}%
9293 }%
```

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.

```
9294 \NeedsTeXFormat{LaTeX2e}
```

Package version number now in line with main glossaries package number but will only be updated when glossaries-accsupp.sty is modified.

```
9295 \ProvidesPackage{glossaries-accsupp}[2014/07/30 v4.08 (NLCT)
9296 Experimental glossaries accessibility]
```

Pass all options to glossaries:

```
9297 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{glossaries}}
```

Process options:

```
9298 \ProcessOptions
```

`\compatibleglossentry` Override style compatibility macros:

```
9299 \def\compatibleglossentry#1#2{%
```

```

9300 \toks@{#2}%
9301 \protected@edef\@do@glossentry{%
9302   \noexpand\accsuppglossaryentryfield{#1}%
9303   {\noexpand\glsnamefont
9304     {\expandafter\expandonce\csname glo@\glsdetoklabel{#1}@name\endcsname}}%
9305   {\expandafter\expandonce\csname glo@\glsdetoklabel{#1}@desc\endcsname}%
9306   {\expandafter\expandonce\csname glo@\glsdetoklabel{#1}@symbol\endcsname}%
9307   {\the\toks@}%
9308 }%
9309 \@do@glossentry
9310 }

```

atiblessubglossentry

```

9311 \def\compatiblessubglossentry#1#2#3{%
9312   \toks@{#3}%
9313   \protected@edef\@do@subglossentry{%
9314     \noexpand\accsuppglossarysubentryfield{\number#1}%
9315     {#2}%
9316     {\noexpand\glsnamefont
9317       {\expandafter\expandonce\csname glo@\glsdetoklabel{#2}@name\endcsname}}%
9318     {\expandafter\expandonce\csname glo@\glsdetoklabel{#2}@desc\endcsname}%
9319     {\expandafter\expandonce\csname glo@\glsdetoklabel{#2}@symbol\endcsname}%
9320     {\the\toks@}%
9321   }%
9322   \@do@subglossentry
9323 }

```

Required packages:

```

9324 \RequirePackage{glossaries}
9325 \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:

```

9326 \define@key{glossentry}{access}{%
9327   \def\@glo@access{#1}%
9328 }

```

textaccess The replacement text corresponding to the text key:

```

9329 \define@key{glossentry}{textaccess}{%
9330   \def\@glo@textaccess{#1}%
9331 }

```

firstaccess The replacement text corresponding to the first key:

```
9332 \define@key{glossentry}{firstaccess}{%
9333   \def\@glo@firstaccess{#1}%
9334 }
```

pluralaccess The replacement text corresponding to the plural key:

```
9335 \define@key{glossentry}{pluralaccess}{%
9336   \def\@glo@pluralaccess{#1}%
9337 }
```

firstpluralaccess The replacement text corresponding to the firstplural key:

```
9338 \define@key{glossentry}{firstpluralaccess}{%
9339   \def\@glo@firstpluralaccess{#1}%
9340 }
```

symbolaccess The replacement text corresponding to the symbol key:

```
9341 \define@key{glossentry}{symbolaccess}{%
9342   \def\@glo@symbolaccess{#1}%
9343 }
```

symbolpluralaccess The replacement text corresponding to the symbolplural key:

```
9344 \define@key{glossentry}{symbolpluralaccess}{%
9345   \def\@glo@symbolpluralaccess{#1}%
9346 }
```

descriptionaccess The replacement text corresponding to the description key:

```
9347 \define@key{glossentry}{descriptionaccess}{%
9348   \def\@glo@descaccess{#1}%
9349 }
```

descriptionpluralaccess The replacement text corresponding to the descriptionplural key:

```
9350 \define@key{glossentry}{descriptionpluralaccess}{%
9351   \def\@glo@descpluralaccess{#1}%
9352 }
```

shortaccess The replacement text corresponding to the short key:

```
9353 \define@key{glossentry}{shortaccess}{%
9354   \def\@glo@shortaccess{#1}%
9355 }
```

shortpluralaccess The replacement text corresponding to the shortplural key:

```
9356 \define@key{glossentry}{shortpluralaccess}{%
9357   \def\@glo@shortpluralaccess{#1}%
9358 }
```

longaccess The replacement text corresponding to the long key:

```
9359 \define@key{glossentry}{longaccess}{%
9360   \def\@glo@longaccess{#1}%
9361 }
```

longpluralaccess The replacement text corresponding to the longplural key:

```
9362 \define@key{glossentry}{longpluralaccess}{%
9363   \def\@glo@longpluralaccess{#1}%
9364 }
```

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:

```
9365 \appto\@gls@keymap{,%
9366   {access}{access},%
9367   {textaccess}{textaccess},%
9368   {firstaccess}{firstaccess},%
9369   {pluralaccess}{pluralaccess},%
9370   {firstpluralaccess}{firstpluralaccess},%
9371   {symbolaccess}{symbolaccess},%
9372   {symbolpluralaccess}{symbolpluralaccess},%
9373   {descaccess}{descaccess},%
9374   {descpluralaccess}{descpluralaccess},%
9375   {shortaccess}{shortaccess},%
9376   {shortpluralaccess}{shortpluralaccess},%
9377   {longaccess}{longaccess},%
9378   {longpluralaccess}{longpluralaccess}%
9379 }
```

\@gls@noaccess Indicates that no replacement text has been provided.

```
9380 \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):

```
9381 \let\@gls@oldnewglossaryentryprehook\@newglossaryentryprehook
9382 \renewcommand*{\@newglossaryentryprehook}{%
9383   \@gls@oldnewglossaryentryprehook
9384   \def\@glo@access{\@glo@symbol}%

```

Initialise the other keys:

```
9385   \def\@glo@textaccess{\@glo@access}%
9386   \def\@glo@firstaccess{\@glo@access}%
9387   \def\@glo@pluralaccess{\@glo@textaccess}%
9388   \def\@glo@firstpluralaccess{\@glo@pluralaccess}%
9389   \def\@glo@symbolaccess{\relax}%
9390   \def\@glo@symbolpluralaccess{\@glo@symbolaccess}%
9391   \def\@glo@descaccess{\relax}%
9392   \def\@glo@descpluralaccess{\@glo@descaccess}%
9393   \def\@glo@shortaccess{\relax}%
9394   \def\@glo@shortpluralaccess{\@glo@shortaccess}%
9395   \def\@glo@longaccess{\relax}%
9396   \def\@glo@longpluralaccess{\@glo@longaccess}%
9397 }
```

Add to the end hook:

```
9398 \let\@gls@oldnewglossaryentryposthook\@newglossaryentryposthook
9399 \renewcommand*{\@newglossaryentryposthook}{%
9400   \@gls@oldnewglossaryentryposthook
```

Store the access information:

```
9401   \expandafter
9402     \protected@xdef\csname glo@\@glo@label @access\endcsname{%
9403       \@glo@access}%
9404   \expandafter
9405     \protected@xdef\csname glo@\@glo@label @textaccess\endcsname{%
9406       \@glo@textaccess}%
9407   \expandafter
9408     \protected@xdef\csname glo@\@glo@label @firstaccess\endcsname{%
9409       \@glo@firstaccess}%
9410   \expandafter
9411     \protected@xdef\csname glo@\@glo@label @pluralaccess\endcsname{%
9412       \@glo@pluralaccess}%
9413   \expandafter
9414     \protected@xdef\csname glo@\@glo@label @firstpluralaccess\endcsname{%
9415       \@glo@firstpluralaccess}%
9416   \expandafter
9417     \protected@xdef\csname glo@\@glo@label @symbolaccess\endcsname{%
9418       \@glo@symbolaccess}%
9419   \expandafter
9420     \protected@xdef\csname glo@\@glo@label @symbolpluralaccess\endcsname{%
9421       \@glo@symbolpluralaccess}%
9422   \expandafter
9423     \protected@xdef\csname glo@\@glo@label @descaccess\endcsname{%
9424       \@glo@descaccess}%
9425   \expandafter
9426     \protected@xdef\csname glo@\@glo@label @descpluralaccess\endcsname{%
9427       \@glo@descpluralaccess}%
9428   \expandafter
9429     \protected@xdef\csname glo@\@glo@label @shortaccess\endcsname{%
9430       \@glo@shortaccess}%
9431   \expandafter
9432     \protected@xdef\csname glo@\@glo@label @shortpluralaccess\endcsname{%
9433       \@glo@shortpluralaccess}%
9434   \expandafter
9435     \protected@xdef\csname glo@\@glo@label @longaccess\endcsname{%
9436       \@glo@longaccess}%
9437   \expandafter
9438     \protected@xdef\csname glo@\@glo@label @longpluralaccess\endcsname{%
9439       \@glo@longpluralaccess}%
9440 }
```


7.2 Accessing Replacement Text

`\glsentryaccess` Get the value of the access key for the entry with the given label:

```
9441 \newcommand*{\glsentryaccess}[1]{%
9442   \@gls@entry@field{#1}{access}%
9443 }
```

`\glsentrytextaccess` Get the value of the textaccess key for the entry with the given label:

```
9444 \newcommand*{\glsentrytextaccess}[1]{%
9445   \@gls@entry@field{#1}{textaccess}%
9446 }
```

`\glsentryfirstaccess` Get the value of the firstaccess key for the entry with the given label:

```
9447 \newcommand*{\glsentryfirstaccess}[1]{%
9448   \@gls@entry@field{#1}{firstaccess}%
9449 }
```

`\glsentrypluralaccess` Get the value of the pluralaccess key for the entry with the given label:

```
9450 \newcommand*{\glsentrypluralaccess}[1]{%
9451   \@gls@entry@field{#1}{pluralaccess}%
9452 }
```

`\glsentryfirstpluralaccess` Get the value of the firstpluralaccess key for the entry with the given label:

```
9453 \newcommand*{\glsentryfirstpluralaccess}[1]{%
9454   \csname glo@#1@firstpluralaccess\endcsname
9455 }
```

`\glsentrysymbolaccess` Get the value of the symbolaccess key for the entry with the given label:

```
9456 \newcommand*{\glsentrysymbolaccess}[1]{%
9457   \@gls@entry@field{#1}{symbolaccess}%
9458 }
```

`\glsentrysymbolpluralaccess` Get the value of the symbolpluralaccess key for the entry with the given label:

```
9459 \newcommand*{\glsentrysymbolpluralaccess}[1]{%
9460   \@gls@entry@field{#1}{symbolpluralaccess}%
9461 }
```

`\glsentrydescaccess` Get the value of the descriptionaccess key for the entry with the given label:

```
9462 \newcommand*{\glsentrydescaccess}[1]{%
9463   \@gls@entry@field{#1}{descaccess}%
9464 }
```

`\glsentrydescpluralaccess` Get the value of the descriptionpluralaccess key for the entry with the given label:

```
9465 \newcommand*{\glsentrydescpluralaccess}[1]{%
9466   \@gls@entry@field{#1}{descaccess}%
9467 }
```

`\glsentryshortaccess` Get the value of the shortaccess key for the entry with the given label:

```
9468 \newcommand*{\glsentryshortaccess}[1]{%
9469   \@gls@entry@field{#1}{shortaccess}%
9470 }
```

`\glsentryshortpluralaccess` Get the value of the shortpluralaccess key for the entry with the given label:

```
9471 \newcommand*{\glsentryshortpluralaccess}[1]{%
9472   \@gls@entry@field{#1}{shortpluralaccess}%
9473 }
```

`\glsentrylongaccess` Get the value of the longaccess key for the entry with the given label:

```
9474 \newcommand*{\glsentrylongaccess}[1]{%
9475   \@gls@entry@field{#1}{longaccess}%
9476 }
```

`\glsentrylongpluralaccess` Get the value of the longpluralaccess key for the entry with the given label:

```
9477 \newcommand*{\glsentrylongpluralaccess}[1]{%
9478   \@gls@entry@field{#1}{longpluralaccess}%
9479 }
```

`\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.)

```
9480 \newcommand*{\glsaccsupp}[2]{%
9481   \BeginAccSupp{ActualText=#1}#2\EndAccSupp{}%
9482 }
```

`\xglsaccsupp` Fully expands replacement text before calling `\glsaccsupp`

```
9483 \newcommand*{\xglsaccsupp}[2]{%
9484   \protected@edef\@gls@replacementtext{#1}%
9485   \expandafter\glsaccsupp\expandafter{\@gls@replacementtext}{#2}%
9486 }
```

`@gls@access@display`

```
9487 \newcommand*{\@gls@access@display}[2]{%
9488   \protected@edef\@glo@access{#2}%
9489   \ifx\@glo@access\@gls@noaccess
9490     #1%
9491   \else
9492     \xglsaccsupp{\@glo@access}{#1}%
9493   \fi
9494 }
```

`\glsnameaccessdisplay` Displays the first argument with the accessibility text for the entry with the label given by the second argument (if set).

```
9495 \DeclareRobustCommand*{\glsnameaccessdisplay}[2]{%
9496   \@gls@access@display{#1}{\glsentryaccess{#2}}%
9497 }
```

lstextaccessdisplay As above but for the textaccess replacement text.

```

9498 \DeclareRobustCommand*\glstextaccessdisplay}[2]{%
9499   \@gls@access@display{#1}{\glsentrytextaccess{#2}}%
9500 }

```

pluralaccessdisplay As above but for the pluralaccess replacement text.

```

9501 \DeclareRobustCommand*\glspluralaccessdisplay}[2]{%
9502   \@gls@access@display{#1}{\glsentrypluralaccess{#2}}%
9503 }

```

sfirstaccessdisplay As above but for the firstaccess replacement text.

```

9504 \DeclareRobustCommand*\glsfirstaccessdisplay}[2]{%
9505   \@gls@access@display{#1}{\glsentryfirstaccess{#2}}%
9506 }

```

pluralaccessdisplay As above but for the firstpluralaccess replacement text.

```

9507 \DeclareRobustCommand*\glsfirstpluralaccessdisplay}[2]{%
9508   \@gls@access@display{#1}{\glsentryfirstpluralaccess{#2}}%
9509 }

```

symbolaccessdisplay As above but for the symbolaccess replacement text.

```

9510 \DeclareRobustCommand*\glssymbolaccessdisplay}[2]{%
9511   \@gls@access@display{#1}{\glsentrysymbolaccess{#2}}%
9512 }

```

pluralaccessdisplay As above but for the symbolpluralaccess replacement text.

```

9513 \DeclareRobustCommand*\glssymbolpluralaccessdisplay}[2]{%
9514   \@gls@access@display{#1}{\glsentrysymbolpluralaccess{#2}}%
9515 }

```

ptionaccessdisplay As above but for the descriptionaccess replacement text.

```

9516 \DeclareRobustCommand*\glsdescriptionaccessdisplay}[2]{%
9517   \@gls@access@display{#1}{\glsentrydescaccess{#2}}%
9518 }

```

pluralaccessdisplay As above but for the descriptionpluralaccess replacement text.

```

9519 \DeclareRobustCommand*\glsdescriptionpluralaccessdisplay}[2]{%
9520   \@gls@access@display{#1}{\glsentrydescpluralaccess{#2}}%
9521 }

```

sshortaccessdisplay As above but for the shortaccess replacement text.

```

9522 \DeclareRobustCommand*\glsshortaccessdisplay}[2]{%
9523   \@gls@access@display{#1}{\glsentryshortaccess{#2}}%
9524 }

```

pluralaccessdisplay As above but for the shortpluralaccess replacement text.

```

9525 \DeclareRobustCommand*\glsshortpluralaccessdisplay}[2]{%
9526   \@gls@access@display{#1}{\glsentryshortpluralaccess{#2}}%
9527 }

```

`longaccessdisplay` As above but for the `longaccess` replacement text.

```

9528 \DeclareRobustCommand*\glslongaccessdisplay}[2]{%
9529   \@gls@access@display{#1}{\glsentrylongaccess{#2}}%
9530 }

```

`pluralaccessdisplay` As above but for the `longpluralaccess` replacement text.

```

9531 \DeclareRobustCommand*\glslongpluralaccessdisplay}[2]{%
9532   \@gls@access@display{#1}{\glsentrylongpluralaccess{#2}}%
9533 }

```

`\glsaccessdisplay` Gets the replacement text corresponding to the named key given by the first argument and calls the appropriate command defined above.

```

9534 \DeclareRobustCommand*\glsaccessdisplay}[3]{%
9535   \@ifundefined{gls#1accessdisplay}%
9536   {%
9537     \PackageError{glossaries-accsupp}{No accessibility support
9538       for key ‘#1’}{%
9539     }%
9540   {%
9541     \csname gls#1accessdisplay\endcsname{#2}{#3}%
9542   }%
9543 }

```

`ls@default@entryfmt` Redefine the default entry format to use accessibility information

```

9544 \renewcommand*\@@gls@default@entryfmt}[2]{%
9545   \ifdefempty\glscustomtext
9546   {%
9547     \glsifplural
9548     {%

```

Plural form

```

9549       \glscapscase
9550       {%

```

Don't adjust case

```

9551       \ifglsused\glslabel
9552       {%

```

Subsequent use

```

9553         #2{\glspluralaccessdisplay
9554           {\glsentryplural{\glslabel}}{\glslabel}}%
9555         {\glsdescriptionpluralaccessdisplay
9556           {\glsentrydescplural{\glslabel}}{\glslabel}}%
9557         {\glssymbolpluralaccessdisplay
9558           {\glsentrysymbolplural{\glslabel}}{\glslabel}}
9559         {\glsinsert}%
9560       }%
9561     }%

```

First use

```

9562      #1{\glsfirstpluralaccessdisplay
9563          {\glsentryfirstplural{\glslabel}}{\glslabel}}%
9564          {\glsdescriptionpluralaccessdisplay
9565              {\glsentrydescplural{\glslabel}}{\glslabel}}%
9566          {\glssymbolpluralaccessdisplay
9567              {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
9568          {\glsinsert}}%
9569      }%
9570  }%
9571  {%

```

Make first letter upper case

```

9572      \ifglsused\glslabel
9573      {%

```

Subsequent use.

```

9574      #2{\glspluralaccessdisplay
9575          {\Glsentryplural{\glslabel}}{\glslabel}}%
9576          {\glsdescriptionpluralaccessdisplay
9577              {\glsentrydescplural{\glslabel}}{\glslabel}}%
9578          {\glssymbolpluralaccessdisplay
9579              {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
9580          {\glsinsert}}%
9581      }%
9582  {%

```

First use

```

9583      #1{\glsfirstpluralaccessdisplay
9584          {\Glsentryfirstplural{\glslabel}}{\glslabel}}%
9585          {\glsdescriptionpluralaccessdisplay
9586              {\glsentrydescplural{\glslabel}}{\glslabel}}%
9587          {\glssymbolpluralaccessdisplay
9588              {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
9589          {\glsinsert}}%
9590      }%
9591  }%
9592  {%

```

Make all upper case

```

9593      \ifglsused\glslabel
9594      {%

```

Subsequent use

```

9595      \MakeUppercase{%
9596          #2{\glspluralaccessdisplay
9597              {\glsentryplural{\glslabel}}{\glslabel}}%
9598              {\glsdescriptionpluralaccessdisplay
9599                  {\glsentrydescplural{\glslabel}}{\glslabel}}%
9600              {\glssymbolpluralaccessdisplay
9601                  {\glsentrysymbolplural{\glslabel}}{\glslabel}}%

```

```

9602          {\glsinsert}}}%
9603      }%
9604      {%

```

First use

```

9605      \MakeUppercase{%
9606          #1{\glsfirstpluralaccessdisplay
9607              {\glsentryfirstplural{\glslabel}}{\glslabel}}}%
9608              {\glsdescriptionpluralaccessdisplay
9609                  {\glsentrydescplural{\glslabel}}{\glslabel}}}%
9610              {\glsymbolpluralaccessdisplay
9611                  {\glsentrysymbolplural{\glslabel}}{\glslabel}}}%
9612              {\glsinsert}}}%
9613      }%
9614      }%
9615      }%
9616      {%

```

Singular form

```

9617      \glscapscase
9618      {%

```

Don't adjust case

```

9619      \ifglsused\glslabel
9620      {%

```

Subsequent use

```

9621      #2{\glstextaccessdisplay
9622          {\glsentrytext{\glslabel}}{\glslabel}}}%
9623          {\glsdescriptionaccessdisplay
9624              {\glsentrydesc{\glslabel}}{\glslabel}}}%
9625              {\glsymbolaccessdisplay
9626                  {\glsentrysymbol{\glslabel}}{\glslabel}}}%
9627              {\glsinsert}}}%
9628      }%
9629      {%

```

First use

```

9630      #1{\glsfirstaccessdisplay
9631          {\glsentryfirst{\glslabel}}{\glslabel}}}%
9632          {\glsdescriptionaccessdisplay
9633              {\glsentrydesc{\glslabel}}{\glslabel}}}%
9634              {\glsymbolaccessdisplay
9635                  {\glsentrysymbol{\glslabel}}{\glslabel}}}%
9636              {\glsinsert}}}%
9637      }%
9638      }%
9639      {%

```

Make first letter upper case

```

9640      \ifglsused\glslabel
9641      {%

```

Subsequent use

```

9642      #2{\glstextaccessdisplay
9643          {\Glsentrytext{\glslabel}}{\glslabel}}%
9644          {\glsdescriptionaccessdisplay
9645              {\Glsentrydesc{\glslabel}}{\glslabel}}%
9646          {\glssymbolaccessdisplay
9647              {\Glsentrysymbol{\glslabel}}{\glslabel}}%
9648          {\glsinsert}}%
9649      }%
9650      {%

```

First use

```

9651      #1{\glsfirstaccessdisplay
9652          {\Glsentryfirst{\glslabel}}{\glslabel}}%
9653          {\glsdescriptionaccessdisplay
9654              {\Glsentrydesc{\glslabel}}{\glslabel}}%
9655          {\glssymbolaccessdisplay
9656              {\Glsentrysymbol{\glslabel}}{\glslabel}}%
9657          {\glsinsert}}%
9658      }%
9659      }%
9660      {%

```

Make all upper case

```

9661      \ifglsused\glslabel
9662      {%

```

Subsequent use

```

9663      \MakeUppercase{%
9664          #2{\glstextaccessdisplay
9665              {\glstextentry{\glslabel}}{\glslabel}}%
9666              {\glsdescriptionaccessdisplay
9667                  {\Glsentrydesc{\glslabel}}{\glslabel}}%
9668              {\glssymbolaccessdisplay
9669                  {\Glsentrysymbol{\glslabel}}{\glslabel}}%
9670              {\glsinsert}}%
9671      }%
9672      {%

```

First use

```

9673      \MakeUppercase{%
9674          #1{\glsfirstaccessdisplay
9675              {\glstextentryfirst{\glslabel}}{\glslabel}}%
9676              {\glsdescriptionaccessdisplay
9677                  {\Glsentrydesc{\glslabel}}{\glslabel}}%
9678              {\glssymbolaccessdisplay
9679                  {\Glsentrysymbol{\glslabel}}{\glslabel}}%
9680              {\glsinsert}}%
9681      }%
9682      }%
9683      }%

```

```

9684 }%
9685 {%

Custom text provided in \glsdisp
9686 \ifglsused{\glslabel}%
9687 {%

Subsequent use
9688 #2{\glscustomtext}%
9689 {\glsdescriptionaccessdisplay
9690 {\glsentrydesc{\glslabel}}{\glslabel}}%
9691 {\glssymbolaccessdisplay
9692 {\glsentrysymbol{\glslabel}}{\glslabel}}%
9693 {\glsinsert}%
9694 }%
9695 {%

First use
9696 #1{\glscustomtext}%
9697 {\glsdescriptionaccessdisplay
9698 {\glsentrydesc{\glslabel}}{\glslabel}}%
9699 {\glssymbolaccessdisplay
9700 {\glsentrysymbol{\glslabel}}{\glslabel}}%
9701 {\glsinsert}%
9702 }%
9703 }%
9704 }

```

`\glsgenentryfmt` Redefine to use accessibility information.

```

9705 \renewcommand*{\glsgenentryfmt}{%
9706 \ifdefempty\glscustomtext
9707 {%
9708 \glsifplural
9709 {%

Plural form
9710 \glscapscase
9711 {%

Don't adjust case
9712 \ifglsused\glslabel
9713 {%

Subsequent use
9714 \glspluralaccessdisplay
9715 {\glsentryplural{\glslabel}}{\glslabel}%
9716 \glsinsert
9717 }%
9718 {%

First use
9719 \glsfirstpluralaccessdisplay

```



```

9720          {\glsentryfirstplural{\glslabel}}{\glslabel}%
9721      \glsinsert
9722  }%
9723  }%
9724  {%

```

Make first letter upper case

```

9725      \ifglsused\glslabel
9726  {%

```

Subsequent use.

```

9727      \glspluralaccessdisplay
9728      {\Glsentryplural{\glslabel}}{\glslabel}%
9729      \glsinsert
9730  }%
9731  {%

```

First use

```

9732      \glsfirstpluralaccessdisplay
9733      {\Glsentryfirstplural{\glslabel}}{\glslabel}%
9734      \glsinsert
9735  }%
9736  }%
9737  {%

```

Make all upper case

```

9738      \ifglsused\glslabel
9739  {%

```

Subsequent use

```

9740      \glspluralaccessdisplay
9741      {\mfirstucMakeUppercase{\glsentryplural{\glslabel}}}%
9742      {\glslabel}%
9743      \mfirstucMakeUppercase{\glsinsert}%
9744  }%
9745  {%

```

First use

```

9746      \glsfirstpluralaccessdisplay
9747      {\mfirstucMakeUppercase{\glsentryfirstplural{\glslabel}}}%
9748      {\glslabel}%
9749      \mfirstucMakeUppercase{\glsinsert}%
9750  }%
9751  }%
9752  }%
9753  {%

```

Singular form

```

9754      \glscapscase
9755  {%

```

Don't adjust case

```

9756      \ifglsused\glslabel
9757      {%

```

Subsequent use

```

9758      \glstextaccessdisplay{\glsentrytext{\glslabel}}{\glslabel}%
9759      \glsinsert
9760      }%
9761      {%

```

First use

```

9762      \glsfirstaccessdisplay{\glsentryfirst{\glslabel}}{\glslabel}%
9763      \glsinsert
9764      }%
9765      }%
9766      {%

```

Make first letter upper case

```

9767      \ifglsused\glslabel
9768      {%

```

Subsequent use

```

9769      \glstextaccessdisplay{\Glsentrytext{\glslabel}}{\glslabel}%
9770      \glsinsert
9771      }%
9772      {%

```

First use

```

9773      \glsfirstaccessdisplay{\Glsentryfirst{\glslabel}}{\glslabel}%
9774      \glsinsert
9775      }%
9776      }%
9777      {%

```

Make all upper case

```

9778      \ifglsused\glslabel
9779      {%

```

Subsequent use

```

9780      \glstextaccessdisplay
9781      {\mfirstucMakeUppercase{\glsentrytext{\glslabel}}}{\glslabel}%
9782      \mfirstucMakeUppercase{\glsinsert}%
9783      }%
9784      {%

```

First use

```

9785      \glsfirstaccessdisplay
9786      {\mfirstucMakeUppercase{\glsentryfirst{\glslabel}}}{\glslabel}%
9787      \mfirstucMakeUppercase{\glsinsert}%
9788      }%
9789      }%
9790      }%
9791      }%
9792      {%

```

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.

```
9793 \glscustomtext\glsinsert
9794 }%
9795 }
```

`\glsngenacfmt` Redefine to include accessibility information.

```
9796 \renewcommand*{\glsngenacfmt}{%
9797 \ifdefempty\glscustomtext
9798 {%
9799 \ifglssused\glslabel
9800 {%
```

Subsequent use:

```
9801 \glsifplural
9802 {%
```

Subsequent plural form:

```
9803 \glscapscase
9804 {%
```

Subsequent plural form, don't adjust case:

```
9805 \acronymfont
9806 {\glsshortpluralaccessdisplay
9807 {\glssentryshortpl{\glslabel}}{\glslabel}}%
9808 \glsinsert
9809 }%
9810 {%
```

Subsequent plural form, make first letter upper case:

```
9811 \acronymfont
9812 {\glsshortpluralaccessdisplay
9813 {\Glsentryshortpl{\glslabel}}{\glslabel}}%
9814 \glsinsert
9815 }%
9816 {%
```

Subsequent plural form, all caps:

```
9817 \mfirstucMakeUppercase
9818 {\acronymfont
9819 {\glsshortpluralaccessdisplay
9820 {\glssentryshortpl{\glslabel}}{\glslabel}}%
9821 \glsinsert}%
9822 }%
9823 }%
9824 {%
```

Subsequent singular form

```
9825 \glscapscase
9826 {%
```

Subsequent singular form, don't adjust case:

```
9827      \acronymfont
9828      {\glsshortaccessdisplay{\glsentryshort{\glslabel}}{\glslabel}}%
9829      \glsinsert
9830      }%
9831      {%
```

Subsequent singular form, make first letter upper case:

```
9832      \acronymfont
9833      {\glsshortaccessdisplay{\Glsentryshort{\glslabel}}{\glslabel}}%
9834      \glsinsert
9835      }%
9836      {%
```

Subsequent singular form, all caps:

```
9837      \mfirstucMakeUppercase
9838      {\acronymfont{%
9839      \glsshortaccessdisplay{\glsentryshort{\glslabel}}{\glslabel}}%
9840      \glsinsert}%
9841      }%
9842      }%
9843      }%
9844      {%
```

First use:

```
9845      \glsifplural
9846      {%
```

First use plural form:

```
9847      \glscapscase
9848      {%
```

First use plural form, don't adjust case:

```
9849      \genplacrfullformat{\glslabel}{\glsinsert}%
9850      }%
9851      {%
```

First use plural form, make first letter upper case:

```
9852      \Genplacrfullformat{\glslabel}{\glsinsert}%
9853      }%
9854      {%
```

First use plural form, all caps:

```
9855      \mfirstucMakeUppercase
9856      {\genplacrfullformat{\glslabel}{\glsinsert}}%
9857      }%
9858      }%
9859      {%
```

First use singular form

```
9860      \glscapscase
9861      {%
```

First use singular form, don't adjust case:

```
9862      \genacrfullformat{\glslabel}{\glsinsert}%
9863      }%
9864      {%
```

First use singular form, make first letter upper case:

```
9865      \Genacrfullformat{\glslabel}{\glsinsert}%
9866      }%
9867      {%
```

First use singular form, all caps:

```
9868      \mfirstucMakeUppercase
9869      {\genacrfullformat{\glslabel}{\glsinsert}}%
9870      }%
9871      }%
9872      }%
9873      }%
9874      {%
```

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.

```
9875      \glscustomtext
9876      }%
9877 }
```

`\genacrfullformat` Redefine to include accessibility information.

```
9878 \renewcommand*{\genacrfullformat}[2]{%
9879   \glslongaccessdisplay{\glsentrylong{#1}}{#1}#2\space
9880   (\glsshortaccessdisplay{\protect\firstacronymfont{\glsentryshort{#1}}}{#1})%
9881 }
```

`\Genacrfullformat` Redefine to include accessibility information.

```
9882 \renewcommand*{\Genacrfullformat}[2]{%
9883   \glslongaccessdisplay{\Glsentrylong{#1}}{#1}#2\space
9884   (\glsshortaccessdisplay{\protect\firstacronymfont{\Glsentryshort{#1}}}{#1})%
9885 }
```

`\genplacrfullformat` Redefine to include accessibility information.

```
9886 \renewcommand*{\genplacrfullformat}[2]{%
9887   \glslongpluralaccessdisplay{\glsentrylongpl{#1}}{#1}#2\space
9888   (\glsshortpluralaccessdisplay
9889     {\protect\firstacronymfont{\glsentryshortpl{#1}}}{#1})%
9890 }
```

`\Genplacrfullformat` Redefine to include accessibility information.

```
9891 \renewcommand*{\Genplacrfullformat}[2]{%
9892   \glslongpluralaccessdisplay{\Glsentrylongpl{#1}}{#1}#2\space
9893   (\glsshortpluralaccessdisplay
9894     {\protect\firstacronymfont{\Glsentryshortpl{#1}}}{#1})%
9895 }
```

\@acrshort

```

9896 \def\@acrshort#1#2[#3]{%
9897   \glsdoifexists{#2}%
9898   {%
9899     \let\do@gl@link@checkfirsthyper\relax

9900     \let\glsifplural\@secondoftwo
9901     \let\glscapscase\@firstofthree
9902     \let\glsinsert\@empty
9903     \def\glscustomtext{%
9904       \acronymfont{\glsshortaccessdisplay{\glsentryshort{#2}}{#2}}#3%
9905     }%

    Call \@gl@link
9906     \@gl@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
9907   }%
9908 }

```

\@Acrshort

```

9909 \def\@Acrshort#1#2[#3]{%
9910   \glsdoifexists{#2}%
9911   {%
9912     \let\do@gl@link@checkfirsthyper\relax

9913     \let\glsifplural\@secondoftwo
9914     \let\glscapscase\@secondofthree
9915     \let\glsinsert\@empty
9916     \def\glscustomtext{%
9917       \acronymfont{\glsshortaccessdisplay{\Glsentryshort{#2}}{#2}}#3%
9918     }%

    Call \@gl@link
9919     \@gl@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
9920   }%
9921 }

```

\@ACRshort

```

9922 \def\@ACRshort#1#2[#3]{%
9923   \glsdoifexists{#2}%
9924   {%
9925     \let\do@gl@link@checkfirsthyper\relax

9926     \let\glsifplural\@secondoftwo
9927     \let\glscapscase\@thirdofthree
9928     \let\glsinsert\@empty
9929     \def\glscustomtext{%
9930       \acronymfont{\glsshortaccessdisplay
9931         {\MakeUppercase{\glsentryshort{#2}}}{#2}}#3%
9932     }%

```

```

Call \@gls@link
9933   \@gls@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
9934   }%
9935 }

```

\@acrlong

```

9936 \def\@acrlong#1#2[#3]{%
9937   \glsdoifexists{#2}%
9938   {%
9939     \let\do@gls@link@checkfirsthyper\relax

9940     \let\glsifplural\@secondoftwo
9941     \let\glscapscase\@firstofthree
9942     \let\glsinsert\@empty
9943     \def\glscustomtext{%
9944       \acronymfont{\glslongaccessdisplay{\glsentrylong{#2}}{#2}}#3%
9945     }%

```

```

Call \@gls@link
9946   \@gls@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
9947   }%
9948 }

```

\@Acrlong

```

9949 \def\@Acrlong#1#2[#3]{%
9950   \glsdoifexists{#2}%
9951   {%
9952     \let\do@gls@link@checkfirsthyper\relax

9953     \let\glsifplural\@secondoftwo
9954     \let\glscapscase\@firstofthree
9955     \let\glsinsert\@empty
9956     \def\glscustomtext{%
9957       \acronymfont{\glslongaccessdisplay{\Glsentrylong{#2}}{#2}}#3%
9958     }%

```

```

Call \@gls@link
9959   \@gls@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
9960   }%
9961 }

```

\@ACRlong

```

9962 \def\@ACRlong#1#2[#3]{%
9963   \glsdoifexists{#2}%
9964   {%
9965     \let\do@gls@link@checkfirsthyper\relax

9966     \let\glsifplural\@secondoftwo
9967     \let\glscapscase\@firstofthree
9968     \let\glsinsert\@empty

```

```

9969 \def\glscustomtext{%
9970 \acronymfont{\glslongaccessdisplay{%
9971 \MakeUppercase{\glsenrylong{#2}}}{#2}#3}%
9972 }%

Call \@gls@link
9973 \@gls@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
9974 }%
9975 }

```

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.

```

9976 \renewcommand*{\glossentryname}[1]{%
9977 \glsdoifexists{#1}%
9978 {%
9979 \glsnamefont{\glsnameaccessdisplay{\glsenryname{#1}}{#1}}%
9980 }%
9981 }

9982 \renewcommand*{\glossentryname}[1]{%
9983 \glsdoifexists{#1}%
9984 {%
9985 \glsnamefont{\glsnameaccessdisplay{\Glsentryname{#1}}{#1}}%
9986 }%
9987 }

9988 \renewcommand*{\glossentrydesc}[1]{%
9989 \glsdoifexists{#1}%
9990 {%
9991 \glsdescriptionaccessdisplay{\glsenrydesc{#1}}{#1}%
9992 }%
9993 }

9994 \renewcommand*{\Glossentrydesc}[1]{%
9995 \glsdoifexists{#1}%
9996 {%
9997 \glsdescriptionaccessdisplay{\Glsentrydesc{#1}}{#1}%
9998 }%
9999 }

10000 \renewcommand*{\glossentrysymbol}[1]{%
10001 \glsdoifexists{#1}%
10002 {%

```



```

10003 \glssymbolaccessdisplay{\glentrysymbol{#1}}{#1}%
10004 }%
10005 }

10006 \renewcommand*{\Glossentrysymbol}[1]{%
10007 \glsoifexists{#1}%
10008 {%
10009 \glssymbolaccessdisplay{\Glentrysymbol{#1}}{#1}%
10010 }%
10011 }

```

pglossaryentryfield

```

10012 \newcommand*{\accsuppglossaryentryfield}[5]{%
10013 \glossaryentryfield{#1}%
10014 {\glnameaccessdisplay{#2}{#1}}%
10015 {\gldescriptionaccessdisplay{#3}{#1}}%
10016 {\glssymbolaccessdisplay{#4}{#1}}{#5}%
10017 }

```

glossarysubentryfield

```

10018 \newcommand*{\accsuppglossarysubentryfield}[6]{%
10019 \glossarysubentryfield{#1}{#2}%
10020 {\glnameaccessdisplay{#3}{#2}}%
10021 {\gldescriptionaccessdisplay{#4}{#2}}%
10022 {\glssymbolaccessdisplay{#5}{#2}}{#6}%
10023 }

```

7.4 Acronyms

Redefine acronym styles provided by glossaries:

long-short *<long>* (*<short>*) acronym style.

```

10024 \renewacronymstyle{long-short}%
10025 {%

```

Check for long form in case this is a mixed glossary.

```

10026 \ifglshaslong{\glslabel}{\glsgenacfmt}{\glsgenentryfmt}%
10027 }%
10028 {%
10029 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
10030 \renewcommand*{\genacrfullformat}[2]{%
10031 \glslongaccessdisplay{\glentrylong{##1}}{##1}##2\space
10032 (\glsshortaccessdisplay
10033 {\protect\firstacronymfont{\glentryshort{##1}}}{##1})%
10034 }%
10035 \renewcommand*{\Genacrfullformat}[2]{%
10036 \glslongaccessdisplay{\Glentrylong{##1}}{##1}##2\space
10037 (\glsshortaccessdisplay
10038 {\protect\firstacronymfont{\glentryshort{##1}}}{##1})%
10039 }%

```

```

10040 \renewcommand*{\genplacrfullformat}[2]{%
10041   \glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1}##2\space
10042   (\glsshortpluralaccessdisplay
10043     {\protect\firstacronymfont{\glsentryshortpl{##1}}}{##1})%
10044 }%
10045 \renewcommand*{\Genplacrfullformat}[2]{%
10046   \glslongpluralaccessdisplay{\Glsentrylongpl{##1}}{##1}##2\space
10047   (\glsshortpluralaccessdisplay
10048     {\protect\firstacronymfont{\Glsentryshortpl{##1}}}{##1})%
10049 }%
10050 \renewcommand*{\acronymentry}[1]{%
10051   \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}}
10052 \renewcommand*{\acronymsort}[2]{##1}%
10053 \renewcommand*{\acronymfont}[1]{##1}%
10054 \renewcommand*{\firstacronymfont}[1]{\acronymfont{##1}}%
10055 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
10056 }

```

short-long <short> (<long>) acronym style.

```

10057 \renewacronymstyle{short-long}%
10058 {%
10059   Check for long form in case this is a mixed glossary.
10060 }%
10061 {%
10062   \ifglshaslong{\glslabel}{\glsgenacfmt}{\glsgenentryfmt}%
10063 }%
10064 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
10065 \renewcommand*{\genacrfullformat}[2]{%
10066   \glsshortaccessdisplay
10067     {\protect\firstacronymfont{\glsentryshort{##1}}}{##1}##2\space
10068   (\glslongaccessdisplay{\glsentrylong{##1}}{##1})%
10069 }%
10070 \renewcommand*{\Genacrfullformat}[2]{%
10071   \glsshortaccessdisplay
10072     {\protect\firstacronymfont{\Glsentryshort{##1}}}{##1}##2\space
10073   (\glslongaccessdisplay{\Glsentrylong{##1}}{##1})%
10074 }%
10075 \renewcommand*{\genplacrfullformat}[2]{%
10076   \glsshortpluralaccessdisplay
10077     {\protect\firstacronymfont{\glsentryshortpl{##1}}}{##1}##2\space
10078   (\glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1})%
10079 }%
10080 \renewcommand*{\Genplacrfullformat}[2]{%
10081   \glsshortpluralaccessdisplay
10082     {\protect\firstacronymfont{\Glsentryshortpl{##1}}}{##1}##2\space
10083   (\glslongpluralaccessdisplay{\Glsentrylongpl{##1}}{##1})%
10084 }%
10085 \renewcommand*{\acronymentry}[1]{%
10086   \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}}%

```

```

10086 \renewcommand*\acronymsort}[2]{##1}%
10087 \renewcommand*\acronymfont}[1]{##1}%
10088 \renewcommand*\firstacronymfont}[1]{\acronymfont{##1}}%
10089 \renewcommand*\acrpluralsuffix{\glspluralsuffix}%
10090 }

```

long-short-desc *<long>* (*<short>*) acronym style that has an accompanying description (which the user needs to supply).

```

10091 \renewacronymstyle{long-short-desc}%
10092 {%
10093   \GlsUseAcrEntryDisplayStyle{long-short}%
10094 }%
10095 {%
10096   \GlsUseAcrStyleDefs{long-short}%
10097   \renewcommand*\GenericAcronymFields{}%
10098   \renewcommand*\acronymsort}[2]{##2}%
10099   \renewcommand*\acronymentry}[1]{%
10100     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10101     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10102 }

```

long-sc-short-desc *<long>* (*\textsc{<short>}*) acronym style that has an accompanying description (which the user needs to supply).

```

10103 \renewacronymstyle{long-sc-short-desc}%
10104 {%
10105   \GlsUseAcrEntryDisplayStyle{long-sc-short}%
10106 }%
10107 {%
10108   \GlsUseAcrStyleDefs{long-sc-short}%
10109   \renewcommand*\GenericAcronymFields{}%
10110   \renewcommand*\acronymsort}[2]{##2}%
10111   \renewcommand*\acronymentry}[1]{%
10112     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10113     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10114 }

```

long-sm-short-desc *<long>* (*\textsmaller{<short>}*) acronym style that has an accompanying description (which the user needs to supply).

```

10115 \renewacronymstyle{long-sm-short-desc}%
10116 {%
10117   \GlsUseAcrEntryDisplayStyle{long-sm-short}%
10118 }%
10119 {%
10120   \GlsUseAcrStyleDefs{long-sm-short}%
10121   \renewcommand*\GenericAcronymFields{}%
10122   \renewcommand*\acronymsort}[2]{##2}%
10123   \renewcommand*\acronymentry}[1]{%
10124     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10125     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%

```

10126 }

short-long-desc *<short>* (*{<long>}*) acronym style that has an accompanying description (which the user needs to supply).

```
10127 \renewacronymstyle{short-long-desc}%
10128 {%
10129   \GlsUseAcrEntryDispStyle{short-long}%
10130 }%
10131 {%
10132   \GlsUseAcrStyleDefs{short-long}%
10133   \renewcommand*{\GenericAcronymFields}{}%
10134   \renewcommand*{\acronymsort}[2]{##2}%
10135   \renewcommand*{\acronymentry}[1]{%
10136     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10137     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10138 }
```

sc-short-long-desc *<long>* (*\textsc{<short>}*) acronym style that has an accompanying description (which the user needs to supply).

```
10139 \renewacronymstyle{sc-short-long-desc}%
10140 {%
10141   \GlsUseAcrEntryDispStyle{sc-short-long}%
10142 }%
10143 {%
10144   \GlsUseAcrStyleDefs{sc-short-long}%
10145   \renewcommand*{\GenericAcronymFields}{}%
10146   \renewcommand*{\acronymsort}[2]{##2}%
10147   \renewcommand*{\acronymentry}[1]{%
10148     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10149     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10150 }
```

sm-short-long-desc *<long>* (*\textsmaller{<short>}*) acronym style that has an accompanying description (which the user needs to supply).

```
10151 \renewacronymstyle{sm-short-long-desc}%
10152 {%
10153   \GlsUseAcrEntryDispStyle{sm-short-long}%
10154 }%
10155 {%
10156   \GlsUseAcrStyleDefs{sm-short-long}%
10157   \renewcommand*{\GenericAcronymFields}{}%
10158   \renewcommand*{\acronymsort}[2]{##2}%
10159   \renewcommand*{\acronymentry}[1]{%
10160     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10161     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10162 }
```

dua *<long>* only acronym style.

10163 \renewacronymstyle{dua}%

10164 {%

Check for long form in case this is a mixed glossary.

10165 \ifdefempty\glscustomtext

10166 {%

10167 \ifglshaslong{\glslabel}%

10168 {%

10169 \glsifplural

10170 {%

Plural form:

10171 \glscapscase

10172 {%

Plural form, don't adjust case:

10173 \glslongpluralaccessdisplay{\glentrylongpl{\glslabel}}{\glslabel}%

10174 \glsinsert

10175 }%

10176 {%

Plural form, make first letter upper case:

10177 \glslongpluralaccessdisplay{\Glsentrylongpl{\glslabel}}{\glslabel}%

10178 \glsinsert

10179 }%

10180 {%

Plural form, all caps:

10181 \glslongpluralaccessdisplay

10182 {\mfirstucMakeUppercase{\glentrylongpl{\glslabel}}}{\glslabel}%

10183 \mfirstucMakeUppercase{\glsinsert}%

10184 }%

10185 }%

10186 {%

Singular form

10187 \glscapscase

10188 {%

Singular form, don't adjust case:

10189 \glslongaccessdisplay{\glentrylong{\glslabel}}{\glslabel}\glsinsert

10190 }%

10191 {%

Subsequent singular form, make first letter upper case:

10192 \glslongaccessdisplay{\Glsentrylong{\glslabel}}{\glslabel}\glsinsert

10193 }%

10194 {%

Subsequent singular form, all caps:

10195 \glslongaccessdisplay

10196 {\mfirstucMakeUppercase

10197 {\glentrylong{\glslabel}\glsinsert}}{\glslabel}%

```

10198         \mfirstucMakeUppercase{\glsinsert}%
10199     }%
10200 }%
10201 }%
10202 {%

    Not an acronym:

10203     \glsgenentryfmt
10204 }%
10205 }%
10206 {\glscustomtext\glsinsert}%
10207 }%
10208 {%
10209     \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
10210     \renewcommand*{\acrfullfmt}[3]{%
10211         \glslink[##1]{##2}{%
10212             \glslongaccessdisplay{\glsentrylong{##2}}{##2}##3\space
10213             (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##2}}}{##2}}}%
10214     \renewcommand*{\Acrfullfmt}[3]{%
10215         \glslink[##1]{##2}{%
10216             \glslongaccessdisplay{\Glsentrylong{##2}}{##2}##3\space
10217             (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##2}}}{##2}}}%
10218     \renewcommand*{\ACRfullfmt}[3]{%
10219         \glslink[##1]{##2}{%
10220             \glslongaccessdisplay
10221             {\mfirstucMakeUppercase{\glsentrylong{##2}}{##2}##3\space
10222             (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##2}}}{##2}}}}}%
10223     \renewcommand*{\acrfullplfmt}[3]{%
10224         \glslink[##1]{##2}{%
10225             \glslongpluralaccessdisplay
10226             {\glsentrylongpl{##2}}{##2}##3\space
10227             (\glsshortpluralaccessdisplay
10228             {\acronymfont{\glsentryshortpl{##2}}}{##2}}}%
10229     \renewcommand*{\Acrfullplfmt}[3]{%
10230         \glslink[##1]{##2}{%
10231             \glslongpluralaccessdisplay
10232             {\Glsentrylongpl{##2}}{##2}##3\space
10233             (\glsshortpluralaccessdisplay
10234             {\acronymfont{\glsentryshortpl{##2}}}{##2}}}%
10235     \renewcommand*{\ACRfullplfmt}[3]{%
10236         \glslink[##1]{##2}{%
10237             \glslongpluralaccessdisplay
10238             {\mfirstucMakeUppercase{\glsentrylongpl{##2}}{##2}##3\space
10239             (\glsshortpluralaccessdisplay
10240             {\acronymfont{\glsentryshortpl{##2}}}{##2}}}}}%
10241     \renewcommand*{\glsentryfull}[1]{%
10242         \glslongaccessdisplay{\glsentrylong{##1}}\space
10243         (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})%
10244     }%
10245     \renewcommand*{\Glsentryfull}[1]{%

```

```

10246 \glslongaccessdisplay{\Glsentrylong{##1}}{##1}\space
10247 (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})%
10248 }%
10249 \renewcommand*{\glsentryfullpl}[1]{%
10250 \glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1}\space
10251 (\glsshortpluralaccessdisplay{\acronymfont{\glsentryshortpl{##1}}}{##1})%
10252 }%
10253 \renewcommand*{\Glsentryfullpl}[1]{%
10254 \glslongpluralaccessdisplay{\Glsentrylongpl{##1}}{##1}\space
10255 (\glsshortpluralaccessdisplay{\acronymfont{\glsentryshortpl{##1}}}{##1})%
10256 }%
10257 \renewcommand*{\acronymentry}[1]{%
10258 \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}%
10259 \renewcommand*{\acronymsort}[2]{##1}%
10260 \renewcommand*{\acronymfont}[1]{##1}%
10261 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
10262 }

```

dua-desc *<long>* only acronym style with user-supplied description.

```

10263 \renewacronymstyle{dua-desc}%
10264 {%
10265 \GlsUseAcrEntryDispStyle{dua}%
10266 }%
10267 {%
10268 \GlsUseAcrStyleDefs{dua}%
10269 \renewcommand*{\GenericAcronymFields}{}%
10270 \renewcommand*{\acronymentry}[1]{%
10271 \glslongaccessdisplay{\acronymfont{\glsentrylong{##1}}}{##1}%
10272 \renewcommand*{\acronymsort}[2]{##2}%
10273 }%

```

footnote *<short>*\footnote{*<long>*} acronym style.

```

10274 \renewacronymstyle{footnote}%
10275 {%

```

Check for long form in case this is a mixed glossary.

```

10276 \ifglshaslong{\glslabel}{\glsngenacfmt}{\glsngenentryfmt}%
10277 }%
10278 }%
10279 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%

```

Need to ensure hyperlinks are switched off on first use:

```

10280 \glshyperfirstfalse
10281 \renewcommand*{\genacrfullformat}[2]{%
10282 \glsshortaccessdisplay
10283 {\protect\firstacronymfont{\glsentryshort{##1}}}{##1}##2%
10284 \protect\footnote{\glslongaccessdisplay{\glsentrylong{##1}}{##1}}%
10285 }%
10286 \renewcommand*{\Genacrfullformat}[2]{%
10287 \glsshortaccessdisplay

```

```

10288     {\firstacronymfont{\Glsentryshort{##1}}}{##1}##2%
10289     \protect\footnote{\glslongaccessdisplay{\glssentrylong{##1}}{##1}}%
10290 }%
10291 \renewcommand*{\genplacrfullformat}[2]{%
10292     \glsshortpluralaccessdisplay
10293     {\protect\firstacronymfont{\glssentryshortpl{##1}}}{##1}##2%
10294     \protect\footnote{\glslongpluralaccessdisplay{\glssentrylongpl{##1}}{##1}}%
10295 }%
10296 \renewcommand*{\Genplacrfullformat}[2]{%
10297     \glsshortpluralaccessdisplay
10298     {\protect\firstacronymfont{\Glsentryshortpl{##1}}}{##1}##2%
10299     \protect\footnote{\glslongpluralaccessdisplay{\glssentrylongpl{##1}}{##1}}%
10300 }%
10301 \renewcommand*{\acronymentry}[1]{%
10302     \glsshortaccessdisplay{\acronymfont{\glssentryshort{##1}}}{##1}}%
10303 \renewcommand*{\acronymsort}[2]{##1}%
10304 \renewcommand*{\acronymfont}[1]{##1}%
10305 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%

```

Don't use footnotes for \acrfull:

```

10306 \renewcommand*{\acrfullfmt}[3]{%
10307     \glslink[##1]{##2}{%
10308         \glsshortaccessdisplay{\acronymfont{\glssentryshort{##2}}}{##2}##3\space
10309         (\glslongaccessdisplay{\glssentrylong{##2}}{##2})}%
10310 \renewcommand*{\Acrfullfmt}[3]{%
10311     \glslink[##1]{##2}{%
10312         \glsshortaccessdisplay{\acronymfont{\Glsentryshort{##2}}}{##2}##3\space
10313         (\glslongaccessdisplay{\glssentrylong{##2}}{##2})}%
10314 \renewcommand*{\ACRfullfmt}[3]{%
10315     \glslink[##1]{##2}{%
10316         \glsshortaccessdisplay
10317         {\mfirstucMakeUppercase
10318         {\acronymfont{\glssentryshort{##2}}}{##2}##3\space
10319         (\glslongaccessdisplay{\glssentrylong{##2}}{##2})}%
10320 \renewcommand*{\acrfullplfmt}[3]{%
10321     \glslink[##1]{##2}{%
10322         \glsshortpluralaccessdisplay
10323         {\acronymfont{\glssentryshortpl{##2}}}{##2}##3\space
10324         (\glslongpluralaccessdisplay{\glssentrylongpl{##2}}{##2})}%
10325 \renewcommand*{\Acrfullplfmt}[3]{%
10326     \glslink[##1]{##2}{%
10327         \glsshortpluralaccessdisplay
10328         {\acronymfont{\Glsentryshortpl{##2}}}{##2}##3\space
10329         (\glslongpluralaccessdisplay{\glssentrylongpl{##2}}{##2})}%
10330 \renewcommand*{\ACRfullplfmt}[3]{%
10331     \glslink[##1]{##2}{%
10332         \glsshortpluralaccessdisplay
10333         {\mfirstucMakeUppercase
10334         {\acronymfont{\glssentryshortpl{##2}}}{##2}##3\space
10335         (\glslongpluralaccessdisplay{\glssentrylongpl{##2}}{##2})}%

```


Similarly for \glsentryfull etc:

```

10336 \renewcommand*{\glsentryfull}[1]{%
10337   \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}\space
10338   (\glslongaccessdisplay{\glsentrylong{##1}}{##1})}%
10339 \renewcommand*{\Glsentryfull}[1]{%
10340   \glsshortaccessdisplay{\acronymfont{\Glsentryshort{##1}}}{##1}\space
10341   (\glslongaccessdisplay{\glsentrylong{##1}}{##1})}%
10342 \renewcommand*{\glsentryfullpl}[1]{%
10343   \glsshortpluralaccessdisplay
10344   {\acronymfont{\glsentryshortpl{##1}}}{##1}\space
10345   (\glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1})}%
10346 \renewcommand*{\Glsentryfullpl}[1]{%
10347   \glsshortpluralaccessdisplay
10348   {\acronymfont{\Glsentryshortpl{##1}}}{##1}\space
10349   (\glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1})}%
10350 }

```

footnote-sc \textsc{<short>}\footnote{<long>} acronym style.

```

10351 \renewacronymstyle{footnote-sc}%
10352 {%
10353   \GlsUseAcrEntryDispStyle{footnote}%
10354 }%
10355 {%
10356   \GlsUseAcrStyleDefs{footnote}%
10357   \renewcommand{\acronymentry}[1]{%
10358     \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}}
10359   \renewcommand{\acronymfont}[1]{\textsc{##1}}%
10360   \renewcommand*{\acrpluralsuffix}{\glstextup{\glspluralsuffix}}%
10361 }%

```

footnote-sm \textsmaller{<short>}\footnote{<long>} acronym style.

```

10362 \renewacronymstyle{footnote-sm}%
10363 {%
10364   \GlsUseAcrEntryDispStyle{footnote}%
10365 }%
10366 {%
10367   \GlsUseAcrStyleDefs{footnote}%
10368   \renewcommand{\acronymentry}[1]{%
10369     \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}}
10370   \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
10371   \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
10372 }%

```

footnote-desc <short>\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```

10373 \renewacronymstyle{footnote-desc}%
10374 {%
10375   \GlsUseAcrEntryDispStyle{footnote}%
10376 }%

```

```

10377 {%
10378   \GlsUseAcrStyleDefs{footnote}%
10379   \renewcommand*{\GenericAcronymFields}{}%
10380   \renewcommand*{\acronymsort}[2]{##2}%
10381   \renewcommand*{\acronymentry}[1]{%
10382     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10383     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10384 }

```

footnote-sc-desc \textsc{\<short>}\footnote{\<long>} acronym style that has an accompanying description (which the user needs to supply).

```

10385 \renewacronymstyle{footnote-sc-desc}%
10386 {%
10387   \GlsUseAcrEntryDispStyle{footnote-sc}%
10388 }%
10389 {%
10390   \GlsUseAcrStyleDefs{footnote-sc}%
10391   \renewcommand*{\GenericAcronymFields}{}%
10392   \renewcommand*{\acronymsort}[2]{##2}%
10393   \renewcommand*{\acronymentry}[1]{%
10394     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10395     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10396 }

```

footnote-sm-desc \textsmaller{\<short>}\footnote{\<long>} acronym style that has an accompanying description (which the user needs to supply).

```

10397 \renewacronymstyle{footnote-sm-desc}%
10398 {%
10399   \GlsUseAcrEntryDispStyle{footnote-sm}%
10400 }%
10401 {%
10402   \GlsUseAcrStyleDefs{footnote-sm}%
10403   \renewcommand*{\GenericAcronymFields}{}%
10404   \renewcommand*{\acronymsort}[2]{##2}%
10405   \renewcommand*{\acronymentry}[1]{%
10406     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10407     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10408 }

```

Use \newacronymhook to modify the key list to set the access text to the long version by default.

```

10409 \renewcommand*{\newacronymhook}{%
10410   \edef\@gls@keylist{shortaccess=\the\glslongtok,%
10411     \the\glskeylisttok}%
10412   \expandafter\glskeylisttok\expandafter{\@gls@keylist}%
10413 }

```

defaultNewAcronymDef Modify default style to use access text:

```

10414 \renewcommand*{\DefaultNewAcronymDef}{%
10415   \edef\@do@newglossaryentry{%
10416     \noexpand\newglossaryentry{\the\glslabeltok}%
10417     {%
10418       type=\acronymtype,%
10419       name={\the\glsshorttok},%
10420       description={\the\glslongtok},%
10421       descriptionaccess=\relax,
10422       text={\the\glsshorttok},%
10423       access={\noexpand\@glo@textaccess},%
10424       sort={\the\glsshorttok},%
10425       short={\the\glsshorttok},%
10426       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10427       shortaccess={\the\glslongtok},%
10428       long={\the\glslongtok},%
10429       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10430       descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10431       first={\noexpand\glslongaccessdisplay
10432         {\the\glslongtok}{\the\glslabeltok}\space
10433         (\noexpand\glsshortaccessdisplay
10434         {\the\glsshorttok}{\the\glslabeltok})},%
10435       plural={\the\glsshorttok\acrpluralsuffix},%
10436       firstplural={\noexpand\glslongpluralaccessdisplay
10437         {\noexpand\@glo@longpl}{\the\glslabeltok}\space
10438         (\noexpand\glsshortpluralaccessdisplay
10439         {\noexpand\@glo@shortpl}{\the\glslabeltok})},%
10440       firstaccess=\relax,
10441       firstpluralaccess=\relax,
10442       textaccess={\noexpand\@glo@shortaccess},%
10443       \the\glskeylisttok
10444     }%
10445   }%
10446   \let\@org@gls@assign@firstpl\gls@assign@firstpl
10447   \let\@org@gls@assign@plural\gls@assign@plural
10448   \let\@org@gls@assign@descplural\gls@assign@descplural
10449   \def\gls@assign@firstpl##1##2{%
10450     \@@gls@expand@field{##1}{firstpl}{##2}%
10451   }%
10452   \def\gls@assign@plural##1##2{%
10453     \@@gls@expand@field{##1}{plural}{##2}%
10454   }%
10455   \def\gls@assign@descplural##1##2{%
10456     \@@gls@expand@field{##1}{descplural}{##2}%
10457   }%
10458   \@do@newglossaryentry
10459   \let\gls@assign@firstpl\@org@gls@assign@firstpl
10460   \let\gls@assign@plural\@org@gls@assign@plural
10461   \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
10462 }

```

otnoteNewAcronymDef

```

10463 \renewcommand*{\DescriptionFootnoteNewAcronymDef}{%
10464   \edef\@do@newglossaryentry{%
10465     \noexpand\newglossaryentry{\the\glslabeltok}%
10466     {%
10467       type=\acronymtype,%
10468       name={\noexpand\acronymfont{\the\glsshorttok}},%
10469       sort={\the\glsshorttok},%
10470       text={\the\glsshorttok},%
10471       short={\the\glsshorttok},%
10472       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10473       shortaccess={\the\glslongtok},%
10474       long={\the\glslongtok},%
10475       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10476       access={\noexpand\@glo@textaccess},%
10477       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10478       symbol={\the\glslongtok},%
10479       symbolplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10480       firstpluralaccess=\relax,
10481       textaccess={\noexpand\@glo@shortaccess},%
10482       \the\glskeylisttok
10483     }%
10484   }%
10485   \let\@org@gls@assign@firstpl\gls@assign@firstpl
10486   \let\@org@gls@assign@plural\gls@assign@plural
10487   \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
10488   \def\gls@assign@firstpl##1##2{%
10489     \@@gls@expand@field{##1}{firstpl}{##2}%
10490   }%
10491   \def\gls@assign@plural##1##2{%
10492     \@@gls@expand@field{##1}{plural}{##2}%
10493   }%
10494   \def\gls@assign@symbolplural##1##2{%
10495     \@@gls@expand@field{##1}{symbolplural}{##2}%
10496   }%
10497   \@do@newglossaryentry
10498   \let\gls@assign@plural\@org@gls@assign@plural
10499   \let\gls@assign@firstpl\@org@gls@assign@firstpl
10500   \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
10501 }

```

ipitionNewAcronymDef

```

10502 \renewcommand*{\DescriptionNewAcronymDef}{%
10503   \edef\@do@newglossaryentry{%
10504     \noexpand\newglossaryentry{\the\glslabeltok}%
10505     {%
10506       type=\acronymtype,%
10507       name={\noexpand
10508         \acrnameformat{\the\glsshorttok}{\the\glslongtok}},%

```

```

10509     access={\noexpand\@glo@textaccess},%
10510     sort={\the\glsshorttok},%
10511     short={\the\glsshorttok},%
10512     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10513     shortaccess={\the\glslongtok},%
10514     long={\the\glslongtok},%
10515     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10516     first={\the\glslongtok},%
10517     firstaccess=\relax,
10518     firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10519     text={\the\glsshorttok},%
10520     textaccess={\the\glslongtok},%
10521     plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10522     symbol={\noexpand\@glo@text},%
10523     symbolaccess={\noexpand\@glo@textaccess},%
10524     symbolplural={\noexpand\@glo@plural},%
10525     firstpluralaccess=\relax,
10526     textaccess={\noexpand\@glo@shortaccess},%
10527     \the\glskeylisttok}%
10528 }%
10529 \let\@org@gls@assign@firstpl\gls@assign@firstpl
10530 \let\@org@gls@assign@plural\gls@assign@plural
10531 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
10532 \def\gls@assign@firstpl##1##2{%
10533   \@@gls@expand@field{##1}{firstpl}{##2}%
10534 }%
10535 \def\gls@assign@plural##1##2{%
10536   \@@gls@expand@field{##1}{plural}{##2}%
10537 }%
10538 \def\gls@assign@symbolplural##1##2{%
10539   \@@gls@expand@field{##1}{symbolplural}{##2}%
10540 }%
10541 \do@newglossaryentry
10542 \let\gls@assign@firstpl\@org@gls@assign@firstpl
10543 \let\gls@assign@plural\@org@gls@assign@plural
10544 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
10545 }

```

otnoteNewAcronymDef

```

10546 \renewcommand*{\FootnoteNewAcronymDef}{%
10547   \edef\@do@newglossaryentry{%
10548     \noexpand\newglossaryentry{\the\glslabeltok}%
10549     {%
10550       type=\acronymtype,%
10551       name={\noexpand\acronymfont{\the\glsshorttok}},%
10552       sort={\the\glsshorttok},%
10553       text={\the\glsshorttok},%
10554       textaccess={\the\glslongtok},%
10555       access={\noexpand\@glo@textaccess},%

```

```

10556 plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10557 short={\the\glsshorttok},%
10558 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10559 long={\the\glslongtok},%
10560 longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10561 description={\the\glslongtok},%
10562 descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10563 \the\glskeylisttok
10564 }%
10565 }%
10566 \let\@org@gls@assign@plural\gls@assign@plural
10567 \let\@org@gls@assign@firstpl\gls@assign@firstpl
10568 \let\@org@gls@assign@descplural\gls@assign@descplural
10569 \def\gls@assign@firstpl##1##2{%
10570   \@@gls@expand@field{##1}{firstpl}{##2}%
10571 }%
10572 \def\gls@assign@plural##1##2{%
10573   \@@gls@expand@field{##1}{plural}{##2}%
10574 }%
10575 \def\gls@assign@descplural##1##2{%
10576   \@@gls@expand@field{##1}{descplural}{##2}%
10577 }%
10578 \@do@newglossaryentry
10579 \let\gls@assign@plural\@org@gls@assign@plural
10580 \let\gls@assign@firstpl\@org@gls@assign@firstpl
10581 \let\gls@assign@descplural\@org@gls@assign@descplural
10582 }

```

\SmallNewAcronymDef

```

10583 \renewcommand*{\SmallNewAcronymDef}{%
10584   \edef\@do@newglossaryentry{%
10585     \noexpand\newglossaryentry{\the\glslabeltok}%
10586     {%
10587       type=\acronymtype,%
10588       name={\noexpand\acronymfont{\the\glsshorttok}},%
10589       access={\noexpand\@glo@symbolaccess},%
10590       sort={\the\glsshorttok},%
10591       short={\the\glsshorttok},%
10592       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10593       shortaccess={\the\glslongtok},%
10594       long={\the\glslongtok},%
10595       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10596       text={\noexpand\@glo@short},%
10597       textaccess={\noexpand\@glo@shortaccess},%
10598       plural={\noexpand\@glo@shortpl},%
10599       first={\the\glslongtok},%
10600       firstaccess=\relax,
10601       firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10602       description={\noexpand\@glo@first},%

```

```

10603      descriptionplural={\noexpand\@glo@firstplural},%
10604      symbol={\the\glsshorttok},%
10605      symbolaccess={\the\glslongtok},%
10606      symbolplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10607      \the\glskeylisttok
10608    }%
10609  }%
10610  \let\@org@gls@assign@firstpl\gls@assign@firstpl
10611  \let\@org@gls@assign@plural\gls@assign@plural
10612  \let\@org@gls@assign@descplural\gls@assign@descplural
10613  \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
10614  \def\gls@assign@firstpl##1##2{%
10615    \@@gls@expand@field{##1}{firstpl}{##2}%
10616  }%
10617  \def\gls@assign@plural##1##2{%
10618    \@@gls@expand@field{##1}{plural}{##2}%
10619  }%
10620  \def\gls@assign@descplural##1##2{%
10621    \@@gls@expand@field{##1}{descplural}{##2}%
10622  }%
10623  \def\gls@assign@symbolplural##1##2{%
10624    \@@gls@expand@field{##1}{symbolplural}{##2}%
10625  }%
10626  \do@newglossaryentry
10627  \let\gls@assign@firstpl\@org@gls@assign@firstpl
10628  \let\gls@assign@plural\@org@gls@assign@plural
10629  \let\gls@assign@descplural\@org@gls@assign@descplural
10630  \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
10631 }

```

The following are kept for compatibility with versions before 3.0:

```

\glsshortaccesskey
10632 \newcommand*{\glsshortaccesskey}{\glsshortkey access}%

hortpluralaccesskey
10633 \newcommand*{\glsshortpluralaccesskey}{\glsshortpluralkey access}%

\glslongaccesskey
10634 \newcommand*{\glslongaccesskey}{\glslongkey access}%

longpluralaccesskey
10635 \newcommand*{\glslongpluralaccesskey}{\glslongpluralkey access}%

```

7.5 Debugging Commands

```

\showglongnameaccess
10636 \newcommand*{\showglongnameaccess}[1]{%
10637   \expandafter\show\curname glo@\glsdetoklabel{#1}@textaccess\endcsname
10638 }

```

```

\showglotextaccess
10639 \newcommand*{\showglotextaccess}[1]{%
10640   \expandafter\show\csname glo@\glsdetoklabel{#1}@textaccess\endcsname
10641 }

showglopluralaccess
10642 \newcommand*{\showglopluralaccess}[1]{%
10643   \expandafter\show\csname glo@\glsdetoklabel{#1}@pluralaccess\endcsname
10644 }

\showglofirstaccess
10645 \newcommand*{\showglofirstaccess}[1]{%
10646   \expandafter\show\csname glo@\glsdetoklabel{#1}@firstaccess\endcsname
10647 }

lofirstpluralaccess
10648 \newcommand*{\showglofirstpluralaccess}[1]{%
10649   \expandafter\show\csname glo@\glsdetoklabel{#1}@firstpluralaccess\endcsname
10650 }

showglosymbolaccess
10651 \newcommand*{\showglosymbolaccess}[1]{%
10652   \expandafter\show\csname glo@\glsdetoklabel{#1}@symbolaccess\endcsname
10653 }

osymbolpluralaccess
10654 \newcommand*{\showglosymbolpluralaccess}[1]{%
10655   \expandafter\show\csname glo@\glsdetoklabel{#1}@symbolpluralaccess\endcsname
10656 }

\showglodescaccess
10657 \newcommand*{\showglodescaccess}[1]{%
10658   \expandafter\show\csname glo@\glsdetoklabel{#1}@descaccess\endcsname
10659 }

glodescpluralaccess
10660 \newcommand*{\showglodescpluralaccess}[1]{%
10661   \expandafter\show\csname glo@\glsdetoklabel{#1}@descpluralaccess\endcsname
10662 }

\showgloshortaccess
10663 \newcommand*{\showgloshortaccess}[1]{%
10664   \expandafter\show\csname glo@\glsdetoklabel{#1}@shortaccess\endcsname
10665 }

loshortpluralaccess
10666 \newcommand*{\showgloshortpluralaccess}[1]{%
10667   \expandafter\show\csname glo@\glsdetoklabel{#1}@shortpluralaccess\endcsname
10668 }

```



```

\showglolongaccess
10669 \newcommand*{\showglolongaccess}[1]{%
10670   \expandafter\show\csname glo@glstetoklabel{#1}@longaccess\endcsname
10671 }

glolongpluralaccess
10672 \newcommand*{\showglolongpluralaccess}[1]{%
10673   \expandafter\show\csname glo@glstetoklabel{#1}@longpluralaccess\endcsname
10674 }

```

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.

```

10675 \NeedsTeXFormat{LaTeX2e}
10676 \ProvidesPackage{glossaries-babel}[2013/11/14 v4.0 (NLCT)]

English:
10677 \@ifundefined{captionsenglish}{-}{%
10678   \addto\captionsenglish{%
10679     \renewcommand*{\glossaryname}{Glossary}%
10680     \renewcommand*{\acronymname}{Acronyms}%
10681     \renewcommand*{\entryname}{Notation}%
10682     \renewcommand*{\descriptionname}{Description}%
10683     \renewcommand*{\symbolname}{Symbol}%
10684     \renewcommand*{\pagelistname}{Page List}%
10685     \renewcommand*{\glssymbolsgroupname}{Symbols}%
10686     \renewcommand*{\glsnumbersgroupname}{Numbers}%
10687 }%
10688 }
10689 \@ifundefined{captionsamerican}{-}{%
10690   \addto\captionsamerican{%
10691     \renewcommand*{\glossaryname}{Glossary}%
10692     \renewcommand*{\acronymname}{Acronyms}%
10693     \renewcommand*{\entryname}{Notation}%
10694     \renewcommand*{\descriptionname}{Description}%
10695     \renewcommand*{\symbolname}{Symbol}%
10696     \renewcommand*{\pagelistname}{Page List}%
10697     \renewcommand*{\glssymbolsgroupname}{Symbols}%
10698     \renewcommand*{\glsnumbersgroupname}{Numbers}%
10699 }%
10700 }
10701 \@ifundefined{captionsaustralian}{-}{%

```

```

10702 \addto\captionsaustrian{%
10703   \renewcommand*{\glossaryname}{Glossary}%
10704   \renewcommand*{\acronymname}{Acronyms}%
10705   \renewcommand*{\entryname}{Notation}%
10706   \renewcommand*{\descriptionname}{Description}%
10707   \renewcommand*{\symbolname}{Symbol}%
10708   \renewcommand*{\pagelistname}{Page List}%
10709   \renewcommand*{\glssymbolsgroupname}{Symbols}%
10710   \renewcommand*{\glsnumbersgroupname}{Numbers}%
10711 }%
10712 }
10713 \@ifundefined{captionsbritish}{-}{%
10714   \addto\captionsbritish{%
10715     \renewcommand*{\glossaryname}{Glossary}%
10716     \renewcommand*{\acronymname}{Acronyms}%
10717     \renewcommand*{\entryname}{Notation}%
10718     \renewcommand*{\descriptionname}{Description}%
10719     \renewcommand*{\symbolname}{Symbol}%
10720     \renewcommand*{\pagelistname}{Page List}%
10721     \renewcommand*{\glssymbolsgroupname}{Symbols}%
10722     \renewcommand*{\glsnumbersgroupname}{Numbers}%
10723 }%
10724 \@ifundefined{captionscanadian}{-}{%
10725   \addto\captionscanadian{%
10726     \renewcommand*{\glossaryname}{Glossary}%
10727     \renewcommand*{\acronymname}{Acronyms}%
10728     \renewcommand*{\entryname}{Notation}%
10729     \renewcommand*{\descriptionname}{Description}%
10730     \renewcommand*{\symbolname}{Symbol}%
10731     \renewcommand*{\pagelistname}{Page List}%
10732     \renewcommand*{\glssymbolsgroupname}{Symbols}%
10733     \renewcommand*{\glsnumbersgroupname}{Numbers}%
10734 }%
10735 }
10736 \@ifundefined{captionsnewzealand}{-}{%
10737   \addto\captionnewzealand{%
10738     \renewcommand*{\glossaryname}{Glossary}%
10739     \renewcommand*{\acronymname}{Acronyms}%
10740     \renewcommand*{\entryname}{Notation}%
10741     \renewcommand*{\descriptionname}{Description}%
10742     \renewcommand*{\symbolname}{Symbol}%
10743     \renewcommand*{\pagelistname}{Page List}%
10744     \renewcommand*{\glssymbolsgroupname}{Symbols}%
10745     \renewcommand*{\glsnumbersgroupname}{Numbers}%
10746 }%
10747 }
10748 \@ifundefined{captionsUKenglish}{-}{%
10749   \addto\captionUKenglish{%
10750     \renewcommand*{\glossaryname}{Glossary}%

```

```

10751 \renewcommand*{\acronymname}{Acronyms}%
10752 \renewcommand*{\entryname}{Notation}%
10753 \renewcommand*{\descriptionname}{Description}%
10754 \renewcommand*{\symbolname}{Symbol}%
10755 \renewcommand*{\pagelistname}{Page List}%
10756 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10757 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10758 }%
10759 }
10760 \@ifundefined{captionsUSenglish}{-}{%
10761 \addto\captionsUSenglish{%
10762 \renewcommand*{\glossaryname}{Glossary}%
10763 \renewcommand*{\acronymname}{Acronyms}%
10764 \renewcommand*{\entryname}{Notation}%
10765 \renewcommand*{\descriptionname}{Description}%
10766 \renewcommand*{\symbolname}{Symbol}%
10767 \renewcommand*{\pagelistname}{Page List}%
10768 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10769 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10770 }%
10771 }

```

German (quite a few variations were suggested for German; I settled on the following):

```

10772 \@ifundefined{captionsgerman}{-}{%
10773 \addto\captionsgerman{%
10774 \renewcommand*{\glossaryname}{Glossar}%
10775 \renewcommand*{\acronymname}{Akronyme}%
10776 \renewcommand*{\entryname}{Bezeichnung}%
10777 \renewcommand*{\descriptionname}{Beschreibung}%
10778 \renewcommand*{\symbolname}{Symbol}%
10779 \renewcommand*{\pagelistname}{Seiten}%
10780 \renewcommand*{\glssymbolsgroupname}{Symbole}%
10781 \renewcommand*{\glsnumbersgroupname}{Zahlen}}
10782 }

```

ngerman is identical to German:

```

10783 \@ifundefined{captionsgerman}{-}{%
10784 \addto\captionsgerman{%
10785 \renewcommand*{\glossaryname}{Glossar}%
10786 \renewcommand*{\acronymname}{Akronyme}%
10787 \renewcommand*{\entryname}{Bezeichnung}%
10788 \renewcommand*{\descriptionname}{Beschreibung}%
10789 \renewcommand*{\symbolname}{Symbol}%
10790 \renewcommand*{\pagelistname}{Seiten}%
10791 \renewcommand*{\glssymbolsgroupname}{Symbole}%
10792 \renewcommand*{\glsnumbersgroupname}{Zahlen}}
10793 }

```

Italian:

```

10794 \@ifundefined{captionsitalian}{-}{%

```

```

10795 \addto\captionstalian{%
10796 \renewcommand*{\glossaryname}{Glossario}%
10797 \renewcommand*{\acronymname}{Acronimi}%
10798 \renewcommand*{\entryname}{Nomenclatura}%
10799 \renewcommand*{\descriptionname}{Descrizione}%
10800 \renewcommand*{\symbolname}{Simbolo}%
10801 \renewcommand*{\pagelistname}{Elenco delle pagine}%
10802 \renewcommand*{\glssymbolsgroupname}{Simboli}%
10803 \renewcommand*{\glsnumbersgroupname}{Numeri}}
10804 }

```

Dutch:

```

10805 \@ifundefined{captionsdutch}{}{%
10806 \addto\captionsdutch{%
10807 \renewcommand*{\glossaryname}{Woordenlijst}%
10808 \renewcommand*{\acronymname}{Acroniemen}%
10809 \renewcommand*{\entryname}{Benaming}%
10810 \renewcommand*{\descriptionname}{Beschrijving}%
10811 \renewcommand*{\symbolname}{Symbool}%
10812 \renewcommand*{\pagelistname}{Pagina's}%
10813 \renewcommand*{\glssymbolsgroupname}{Symbolen}%
10814 \renewcommand*{\glsnumbersgroupname}{Cijfers}}
10815 }

```

Spanish:

```

10816 \@ifundefined{captionsspanish}{}{%
10817 \addto\captionsspanish{%
10818 \renewcommand*{\glossaryname}{Glosario}%
10819 \renewcommand*{\acronymname}{Siglas}%
10820 \renewcommand*{\entryname}{Entrada}%
10821 \renewcommand*{\descriptionname}{Descripci\'on}%
10822 \renewcommand*{\symbolname}{S\'mbolo}%
10823 \renewcommand*{\pagelistname}{Lista de p\'aginas}%
10824 \renewcommand*{\glssymbolsgroupname}{S\'mbolos}%
10825 \renewcommand*{\glsnumbersgroupname}{N\'umeros}}
10826 }

```

French:

```

10827 \@ifundefined{captionsfrench}{}{%
10828 \addto\captionsfrench{%
10829 \renewcommand*{\glossaryname}{Glossaire}%
10830 \renewcommand*{\acronymname}{Acronymes}%
10831 \renewcommand*{\entryname}{Termes}%
10832 \renewcommand*{\descriptionname}{Description}%
10833 \renewcommand*{\symbolname}{Symbole}%
10834 \renewcommand*{\pagelistname}{Pages}%
10835 \renewcommand*{\glssymbolsgroupname}{Symboles}%
10836 \renewcommand*{\glsnumbersgroupname}{Nombres}}
10837 }
10838 \@ifundefined{captionsfrenchb}{}{%
10839 \addto\captionsfrenchb{%

```

```

10840 \renewcommand*{\glossaryname}{Glossaire}%
10841 \renewcommand*{\acronymname}{Acronymes}%
10842 \renewcommand*{\entryname}{Terme}%
10843 \renewcommand*{\descriptionname}{Description}%
10844 \renewcommand*{\symbolname}{Symbole}%
10845 \renewcommand*{\pagelistname}{Pages}%
10846 \renewcommand*{\glssymbolsgroupname}{Symboles}%
10847 \renewcommand*{\glsnumbersgroupname}{Nombres}}
10848 }
10849 \@ifundefined{captionsfrancais}{}{%
10850 \addto\captionsfrancais{%
10851 \renewcommand*{\glossaryname}{Glossaire}%
10852 \renewcommand*{\acronymname}{Acronymes}%
10853 \renewcommand*{\entryname}{Terme}%
10854 \renewcommand*{\descriptionname}{Description}%
10855 \renewcommand*{\symbolname}{Symbole}%
10856 \renewcommand*{\pagelistname}{Pages}%
10857 \renewcommand*{\glssymbolsgroupname}{Symboles}%
10858 \renewcommand*{\glsnumbersgroupname}{Nombres}}
10859 }

```

Danish:

```

10860 \@ifundefined{captionsdanish}{}{%
10861 \addto\captionsdanish{%
10862 \renewcommand*{\glossaryname}{Ordliste}%
10863 \renewcommand*{\acronymname}{Akronymer}%
10864 \renewcommand*{\entryname}{Symbolforklaring}%
10865 \renewcommand*{\descriptionname}{Beskrivelse}%
10866 \renewcommand*{\symbolname}{Symbol}%
10867 \renewcommand*{\pagelistname}{Side}%
10868 \renewcommand*{\glssymbolsgroupname}{Symboler}%
10869 \renewcommand*{\glsnumbersgroupname}{Tal}}
10870 }

```

Irish:

```

10871 \@ifundefined{captionsirish}{}{%
10872 \addto\captionsirish{%
10873 \renewcommand*{\glossaryname}{Gluais}%
10874 \renewcommand*{\acronymname}{Acrainmneacha}%

```

wasn't sure whether to go for Nóta (Note), Ciall ('Meaning', 'sense') or Brí ('Meaning'). In the end I chose Ciall.

```

10875 \renewcommand*{\entryname}{Ciall}%
10876 \renewcommand*{\descriptionname}{Tuairisc}%

```

Again, not sure whether to use Comhartha/Comharthaí or Siombail/Siombaile, so have chosen the former.

```

10877 \renewcommand*{\symbolname}{Comhartha}%
10878 \renewcommand*{\glssymbolsgroupname}{Comhartha\'}{\i}}%
10879 \renewcommand*{\pagelistname}{Leathanaigh}%
10880 \renewcommand*{\glsnumbersgroupname}{Uimhreacha}}

```

10881 }

Hungarian:

```
10882 \@ifundefined{captionsmagyar}{%{}%
10883   \addto\captionsmagyar{%
10884     \renewcommand*{\glossaryname}{Sz\`ojegyz\`ek}%
10885     \renewcommand*{\acronymname}{Bet\H uszavak}%
10886     \renewcommand*{\entryname}{Kifejez\`es}%
10887     \renewcommand*{\descriptionname}{Magyar\`azat}%
10888     \renewcommand*{\symbolname}{Jel\"ol\`es}%
10889     \renewcommand*{\pagelistname}{Oldalsz\`am}%
10890     \renewcommand*{\glssymbolsgroupname}{Jelek}%
10891     \renewcommand*{\glsnumbersgroupname}{Sz\`amjegyek}%
10892   }
10893 }
10894 \@ifundefined{captionshungarian}{%{}%
10895   \addto\captionshungarian{%
10896     \renewcommand*{\glossaryname}{Sz\`ojegyz\`ek}%
10897     \renewcommand*{\acronymname}{Bet\H uszavak}%
10898     \renewcommand*{\entryname}{Kifejez\`es}%
10899     \renewcommand*{\descriptionname}{Magyar\`azat}%
10900     \renewcommand*{\symbolname}{Jel\"ol\`es}%
10901     \renewcommand*{\pagelistname}{Oldalsz\`am}%
10902     \renewcommand*{\glssymbolsgroupname}{Jelek}%
10903     \renewcommand*{\glsnumbersgroupname}{Sz\`amjegyek}%
10904   }
10905 }
```

Polish

```
10906 \@ifundefined{captionspolish}{%{}%
10907   \addto\captionspolish{%
10908     \renewcommand*{\glossaryname}{S{\l}ownik termin\`ow}%
10909     \renewcommand*{\acronymname}{Skr\`ot}%
10910     \renewcommand*{\entryname}{Termin}%
10911     \renewcommand*{\descriptionname}{Opis}%
10912     \renewcommand*{\symbolname}{Symbol}%
10913     \renewcommand*{\pagelistname}{Strony}%
10914     \renewcommand*{\glssymbolsgroupname}{Symbole}%
10915     \renewcommand*{\glsnumbersgroupname}{Liczby}}
10916 }
```

Brazilian

```
10917 \@ifundefined{captionsbrazil}{%{}%
10918   \addto\captionsbrazil{%
10919     \renewcommand*{\glossaryname}{Gloss\`ario}%
10920     \renewcommand*{\acronymname}{Siglas}%
10921     \renewcommand*{\entryname}{Nota\c c\~ao}%
10922     \renewcommand*{\descriptionname}{Descri\c c\~ao}%
10923     \renewcommand*{\symbolname}{S\`imbolo}%
10924     \renewcommand*{\pagelistname}{Lista de P\`aginas}%
10925     \renewcommand*{\glssymbolsgroupname}{S\`imbolos}%

```

```

10926 \renewcommand*{\glsnumbersgroupname}{N\,umeros}%
10927 }%
10928 }

```

8.2 Polyglossia Captions

```

10929 \NeedsTeXFormat{LaTeX2e}
10930 \ProvidesPackage{glossaries-polyglossia}[2013/11/14 v4.0 (NLCT)]

```

English:

```

10931 \@ifundefined{captionsenglish}{}{%
10932 \expandafter\toks@\expandafter{\captionsenglish
10933 \renewcommand*{\glossaryname}{\textenglish{Glossary}}%
10934 \renewcommand*{\acronymname}{\textenglish{Acronyms}}%
10935 \renewcommand*{\entryname}{\textenglish{Notation}}%
10936 \renewcommand*{\descriptionname}{\textenglish{Description}}%
10937 \renewcommand*{\symbolname}{\textenglish{Symbol}}%
10938 \renewcommand*{\pagelistname}{\textenglish{Page List}}%
10939 \renewcommand*{\glssymbolsgroupname}{\textenglish{Symbols}}%
10940 \renewcommand*{\glsnumbersgroupname}{\textenglish{Numbers}}%
10941 }%
10942 \edef\captionsenglish{\the\toks@}%
10943 }

```

German:

```

10944 \@ifundefined{captionsgerman}{}{%
10945 \expandafter\toks@\expandafter{\captionsgerman
10946 \renewcommand*{\glossaryname}{\textgerman{Glossar}}%
10947 \renewcommand*{\acronymname}{\textgerman{Akronyme}}%
10948 \renewcommand*{\entryname}{\textgerman{Bezeichnung}}%
10949 \renewcommand*{\descriptionname}{\textgerman{Beschreibung}}%
10950 \renewcommand*{\symbolname}{\textgerman{Symbol}}%
10951 \renewcommand*{\pagelistname}{\textgerman{Seiten}}%
10952 \renewcommand*{\glssymbolsgroupname}{\textgerman{Symbole}}%
10953 \renewcommand*{\glsnumbersgroupname}{\textgerman{Zahlen}}%
10954 }%
10955 \edef\captionsgerman{\the\toks@}%
10956 }

```

Italian:

```

10957 \@ifundefined{captionsitalian}{}{%
10958 \expandafter\toks@\expandafter{\captionsitalian
10959 \renewcommand*{\glossaryname}{\textitalian{Glossario}}%
10960 \renewcommand*{\acronymname}{\textitalian{Acronimi}}%
10961 \renewcommand*{\entryname}{\textitalian{Nomenclatura}}%
10962 \renewcommand*{\descriptionname}{\textitalian{Descrizione}}%
10963 \renewcommand*{\symbolname}{\textitalian{Simbolo}}%
10964 \renewcommand*{\pagelistname}{\textitalian{Elenco delle pagine}}%
10965 \renewcommand*{\glssymbolsgroupname}{\textitalian{Simboli}}%
10966 \renewcommand*{\glsnumbersgroupname}{\textitalian{Numeri}}%
10967 }%

```

```

10968 \edef\captionsitalian{\the\toks@}%
10969 }

```

Dutch:

```

10970 \@ifundefined{captionsdutch}{}{%
10971 \expandafter\toks@\expandafter{\captionsdutch
10972 \renewcommand*{\glossaryname}{\textdutch{Woordenlijst}}%
10973 \renewcommand*{\acronymname}{\textdutch{Acroniemen}}%
10974 \renewcommand*{\entryname}{\textdutch{Benaming}}%
10975 \renewcommand*{\descriptionname}{\textdutch{Beschrijving}}%
10976 \renewcommand*{\symbolname}{\textdutch{Symbool}}%
10977 \renewcommand*{\pagelistname}{\textdutch{Pagina's}}%
10978 \renewcommand*{\glssymbolsgroupname}{\textdutch{Symbolen}}%
10979 \renewcommand*{\glsnumbersgroupname}{\textdutch{Cijfers}}%
10980 }%
10981 \edef\captionsdutch{\the\toks@}%
10982 }

```

Spanish:

```

10983 \@ifundefined{captionsspanish}{}{%
10984 \expandafter\toks@\expandafter{\captionsspanish
10985 \renewcommand*{\glossaryname}{\textspanish{Glosario}}%
10986 \renewcommand*{\acronymname}{\textspanish{Siglas}}%
10987 \renewcommand*{\entryname}{\textspanish{Entrada}}%
10988 \renewcommand*{\descriptionname}{\textspanish{Descripci'on}}%
10989 \renewcommand*{\symbolname}{\textspanish{S'\i mbolo}}%
10990 \renewcommand*{\pagelistname}{\textspanish{Lista de p'aginas}}%
10991 \renewcommand*{\glssymbolsgroupname}{\textspanish{S'\i mbolos}}%
10992 \renewcommand*{\glsnumbersgroupname}{\textspanish{N'umeros}}%
10993 }%
10994 \edef\captionsspanish{\the\toks@}%
10995 }

```

French:

```

10996 \@ifundefined{captionsfrench}{}{%
10997 \expandafter\toks@\expandafter{\captionsfrench
10998 \renewcommand*{\glossaryname}{\textfrench{Glossaire}}%
10999 \renewcommand*{\acronymname}{\textfrench{Acronymes}}%
11000 \renewcommand*{\entryname}{\textfrench{Terme}}%
11001 \renewcommand*{\descriptionname}{\textfrench{Description}}%
11002 \renewcommand*{\symbolname}{\textfrench{Symbole}}%
11003 \renewcommand*{\pagelistname}{\textfrench{Pages}}%
11004 \renewcommand*{\glssymbolsgroupname}{\textfrench{Symboles}}%
11005 \renewcommand*{\glsnumbersgroupname}{\textfrench{Nombres}}%
11006 }%
11007 \edef\captionsfrench{\the\toks@}%
11008 }

```

Danish:

```

11009 \@ifundefined{captionsdanish}{}{%
11010 \expandafter\toks@\expandafter{\captionsdanish
11011 \renewcommand*{\glossaryname}{\textdanish{Ordliste}}%

```



```

11012 \renewcommand*{\acronymname}{\textdanish{Akronymer}}}%
11013 \renewcommand*{\entryname}{\textdanish{Symbolforklaring}}}%
11014 \renewcommand*{\descriptionname}{\textdanish{Beskrivelse}}}%
11015 \renewcommand*{\symbolname}{\textdanish{Symbol}}}%
11016 \renewcommand*{\pagelistname}{\textdanish{Side}}}%
11017 \renewcommand*{\glssymbolsgroupname}{\textdanish{Symboler}}}%
11018 \renewcommand*{\glslnumbersgroupname}{\textdanish{Tal}}}%
11019 }%
11020 \edef\captionsdanish{\the\toks@}%
11021 }

Irish:
11022 \@ifundefined{captionsirish}{}{%
11023 \expandafter\toks@\expandafter{\captionsirish
11024 \renewcommand*{\glossaryname}{\textirish{Gluais}}}%
11025 \renewcommand*{\acronymname}{\textirish{Acrainmneacha}}}%
11026 \renewcommand*{\entryname}{\textirish{Ciall}}}%
11027 \renewcommand*{\descriptionname}{\textirish{Tuirisc}}}%
11028 \renewcommand*{\symbolname}{\textirish{Comhartha}}}%
11029 \renewcommand*{\glssymbolsgroupname}{\textirish{Comhartha\`{\i}}}%
11030 \renewcommand*{\pagelistname}{\textirish{Leathanaigh}}}%
11031 \renewcommand*{\glslnumbersgroupname}{\textirish{Uimhreacha}}}%
11032 }%
11033 \edef\captionsirish{\the\toks@}%
11034 }

Hungarian:
11035 \@ifundefined{captionsmagyar}{}{%
11036 \expandafter\toks@\expandafter{\captionsmagyar
11037 \renewcommand*{\glossaryname}{\textmagyar{Sz\`ojegyz\`ek}}}%
11038 \renewcommand*{\acronymname}{\textmagyar{Bet\H uszavak}}}%
11039 \renewcommand*{\entryname}{\textmagyar{Kifejez\`es}}}%
11040 \renewcommand*{\descriptionname}{\textmagyar{Magyar\`azat}}}%
11041 \renewcommand*{\symbolname}{\textmagyar{Jel\`ol\`es}}}%
11042 \renewcommand*{\pagelistname}{\textmagyar{Oldalsz\`am}}}%
11043 \renewcommand*{\glssymbolsgroupname}{\textmagyar{Jelek}}}%
11044 \renewcommand*{\glslnumbersgroupname}{\textmagyar{Sz\`amjegyek}}}%
11045 }%
11046 \edef\captionsmagyar{\the\toks@}%
11047 }

Polish
11048 \@ifundefined{captionspolish}{}{%
11049 \expandafter\toks@\expandafter{\captionspolish
11050 \renewcommand*{\glossaryname}{\textpolish{S{\l}ownik termin\`ow}}}%
11051 \renewcommand*{\acronymname}{\textpolish{Skr\`ot}}}%
11052 \renewcommand*{\entryname}{\textpolish{Termin}}}%
11053 \renewcommand*{\descriptionname}{\textpolish{Opis}}}%
11054 \renewcommand*{\symbolname}{\textpolish{Symbol}}}%
11055 \renewcommand*{\pagelistname}{\textpolish{Strony}}}%
11056 \renewcommand*{\glssymbolsgroupname}{\textpolish{Symbole}}}%

```

```

11057 \renewcommand*{\glsnumbersgroupname}{\textpolish{Liczby}}%
11058 }%
11059 \edef\captionspolish{\the\toks@}%
11060 }

```

Portugues

```

11061 \@ifundefined{captionsportuges}{}{%
11062 \expandafter\toks@\expandafter{\captionsportuges
11063 \renewcommand*{\glossaryname}{\textportuges{Gloss\'ario}}%
11064 \renewcommand*{\acronymname}{\textportuges{Siglas}}%
11065 \renewcommand*{\entryname}{\textportuges{Nota\c c\~ao}}%
11066 \renewcommand*{\descriptionname}{\textportuges{Descri\c c\~ao}}%
11067 \renewcommand*{\symbolname}{\textportuges{S\'imbolo}}%
11068 \renewcommand*{\pagelistname}{\textportuges{Lista de P\'aginas}}%
11069 \renewcommand*{\glssymbolsgroupname}{\textportuges{S\'imbolos}}%
11070 \renewcommand*{\glsnumbersgroupname}{\textportuges{N\'umeros}}%
11071 }%
11072 \edef\captionsportuges{\the\toks@}%
11073 }

```

8.3 Brazilian Dictionary

This is a dictionary file provided by Thiago de Melo for use with the package.

```

11074 \ProvidesDictionary{glossaries-dictionary}{Brazilian}

```

Provide Brazilian translations:

```

11075 \providetranslation{Glossary}{Gloss\'ario}
11076 \providetranslation{Acronyms}{Siglas}
11077 \providetranslation{Notation (glossaries)}{Nota\c c\~ao}
11078 \providetranslation{Description (glossaries)}{Descri\c c\~ao}
11079 \providetranslation{Symbol (glossaries)}{S\'imbolo}
11080 \providetranslation{Page List (glossaries)}{Lista de P\'aginas}
11081 \providetranslation{Symbols (glossaries)}{S\'imbolos}
11082 \providetranslation{Numbers (glossaries)}{N\'umeros}

```

8.4 Danish Dictionary

This is a dictionary file provided for use with the package.

```

11083 \ProvidesDictionary{glossaries-dictionary}{Danish}

```

Provide Danish translations:

```

11084 \providetranslation{Glossary}{Ordliste}
11085 \providetranslation{Acronyms}{Akronymer}
11086 \providetranslation{Notation (glossaries)}{Symbolforklaring}
11087 \providetranslation{Description (glossaries)}{Beskrivelse}
11088 \providetranslation{Symbol (glossaries)}{Symbol}
11089 \providetranslation{Page List (glossaries)}{Side}
11090 \providetranslation{Symbols (glossaries)}{Symboler}
11091 \providetranslation{Numbers (glossaries)}{Tal}

```

8.5 Dutch Dictionary

This is a dictionary file provided for use with the package.

```
11092 \ProvidesDictionary{glossaries-dictionary}{Dutch}
```

Provide Dutch translations:

```
11093 \providetranslation{Glossary}{Woordenlijst}
11094 \providetranslation{Acronyms}{Acroniemen}
11095 \providetranslation{Notation (glossaries)}{Benaming}
11096 \providetranslation{Description (glossaries)}{Beschrijving}
11097 \providetranslation{Symbol (glossaries)}{Symbool}
11098 \providetranslation{Page List (glossaries)}{Pagina's}
11099 \providetranslation{Symbols (glossaries)}{Symbolen}
11100 \providetranslation{Numbers (glossaries)}{Cijfers}
```

8.6 English Dictionary

This is a dictionary file provided for use with the package.

```
11101 \ProvidesDictionary{glossaries-dictionary}{English}
```

Provide English translations:

```
11102 \providetranslation{Glossary}{Glossary}
11103 \providetranslation{Acronyms}{Acronyms}
11104 \providetranslation{Notation (glossaries)}{Notation}
11105 \providetranslation{Description (glossaries)}{Description}
11106 \providetranslation{Symbol (glossaries)}{Symbol}
11107 \providetranslation{Page List (glossaries)}{Page List}
11108 \providetranslation{Symbols (glossaries)}{Symbols}
11109 \providetranslation{Numbers (glossaries)}{Numbers}
```

8.7 French Dictionary

This is a dictionary file provided for use with the package.

```
11110 \ProvidesDictionary{glossaries-dictionary}{French}
```

Provide French translations:

```
11111 \providetranslation{Glossary}{Glossaire}
11112 \providetranslation{Acronyms}{Acronymes}
11113 \providetranslation{Notation (glossaries)}{Terme}
11114 \providetranslation{Description (glossaries)}{Description}
11115 \providetranslation{Symbol (glossaries)}{Symbole}
11116 \providetranslation{Page List (glossaries)}{Pages}
11117 \providetranslation{Symbols (glossaries)}{Symboles}
11118 \providetranslation{Numbers (glossaries)}{Nombres}
```

8.8 German Dictionary

This is a dictionary file provided for use with the package.

```
11119 \ProvidesDictionary{glossaries-dictionary}{German}
```

Provide German translations (quite a few variations were suggested for German; I settled on the following):

```
11120 \providetranslation{Glossary}{Glossar}
11121 \providetranslation{Acronyms}{Akronyme}
11122 \providetranslation{Notation (glossaries)}{Bezeichnung}
11123 \providetranslation{Description (glossaries)}{Beschreibung}
11124 \providetranslation{Symbol (glossaries)}{Symbol}
11125 \providetranslation{Page List (glossaries)}{Seiten}
11126 \providetranslation{Symbols (glossaries)}{Symbole}
11127 \providetranslation{Numbers (glossaries)}{Zahlen}
```

8.9 Irish Dictionary

This is a dictionary file provided for use with the package.

```
11128 \ProvidesDictionary{glossaries-dictionary}{Irish}
```

Provide Irish translations:

```
11129 \providetranslation{Glossary}{Gluais}
11130 \providetranslation{Acronyms}{Acrainmneacha}
11131 \providetranslation{Notation (glossaries)}{Ciall}
11132 \providetranslation{Description (glossaries)}{Tuairisc}
11133 \providetranslation{Symbol (glossaries)}{Comhartha}
11134 \providetranslation{Page List (glossaries)}{Leathanaigh}
11135 \providetranslation{Symbols (glossaries)}{Comhartha'\{i}}
11136 \providetranslation{Numbers (glossaries)}{Uimhreacha}
```

8.10 Italian Dictionary

This is a dictionary file provided for use with the package.

```
11137 \ProvidesDictionary{glossaries-dictionary}{Italian}
```

Provide Italian translations:

```
11138 \providetranslation{Glossary}{Glossario}
11139 \providetranslation{Acronyms}{Acronimi}
11140 \providetranslation{Notation (glossaries)}{Nomenclatura}
11141 \providetranslation{Description (glossaries)}{Descrizione}
11142 \providetranslation{Symbol (glossaries)}{Simbolo}
11143 \providetranslation{Page List (glossaries)}{Elenco delle pagine}
11144 \providetranslation{Symbols (glossaries)}{Simboli}
11145 \providetranslation{Numbers (glossaries)}{Numeri}
```

8.11 Magyar Dictionary

This is a dictionary file provided for use with the package.

```
11146 \ProvidesDictionary{glossaries-dictionary}{Magyar}
```

Provide translations:

```
11147 \providetranslation{Glossary}{Sz\'ojegy\'ek}
11148 \providetranslation{Acronyms}{Bet\'H uszavak}
```

```

11149 \providetranslation{Notation (glossaries)}{Kifejez\`es}
11150 \providetranslation{Description (glossaries)}{Magyar\`azat}
11151 \providetranslation{Symbol (glossaries)}{Jel\`ol\`es}
11152 \providetranslation{Page List (glossaries)}{Oldalsz\`am}
11153 \providetranslation{Symbols (glossaries)}{Jelek}
11154 \providetranslation{Numbers (glossaries)}{Sz\`amjegyek}

```

8.12 Polish Dictionary

This is a dictionary file provided for use with the package.

```

11155 \ProvidesDictionary{glossaries-dictionary}{Polish}

```

Provide Polish translations:

```

11156 \providetranslation{Glossary}{S\`ownik termin\`ow}
11157 \providetranslation{Acronyms}{Skr\`ot}
11158 \providetranslation{Notation (glossaries)}{Termin}
11159 \providetranslation{Description (glossaries)}{Opis}
11160 \providetranslation{Symbol (glossaries)}{Symbol}
11161 \providetranslation{Page List (glossaries)}{Strony}
11162 \providetranslation{Symbols (glossaries)}{Symbole}
11163 \providetranslation{Numbers (glossaries)}{Liczby}

```

8.13 Serbian Dictionary

This dictionary was provided by Zoran Filipovic.

```

11164 \ProvidesDictionary{glossaries-dictionary}{Serbian}
11165 \providetranslation{Glossary}{Mali re\`v cnik}
11166 \providetranslation{Acronyms}{Skra\` cenice}
11167 \providetranslation{Notation (glossaries)}{Oznaka}
11168 \providetranslation{Description (glossaries)}{Opis}
11169 \providetranslation{Symbol (glossaries)}{Simbol}
11170 \providetranslation{Page List (glossaries)}{Stranica}
11171 \providetranslation{Symbols (glossaries)}{Simboli}
11172 \providetranslation{Numbers (glossaries)}{Brojevi}

```

8.14 Spanish Dictionary

This is a dictionary file provided for use with the package.

```

11173 \ProvidesDictionary{glossaries-dictionary}{Spanish}

```

Provide Spanish translations:

```

11174 \providetranslation{Glossary}{Glosario}
11175 \providetranslation{Acronyms}{Siglas}
11176 \providetranslation{Notation (glossaries)}{Entrada}
11177 \providetranslation{Description (glossaries)}{Descripci\`on}
11178 \providetranslation{Symbol (glossaries)}{S\`mbolo}
11179 \providetranslation{Page List (glossaries)}{Lista de p\`aginas}
11180 \providetranslation{Symbols (glossaries)}{S\`mbolos}
11181 \providetranslation{Numbers (glossaries)}{N\`umeros}

```

Glossary

`makeindex` An indexing application. 10, 23, 24, 158

`xindy` An flexible indexing application with multilingual support written in Perl. 10, 23, 24, 158

Change History

??		<code>\capitalisewords</code> : made robust	243
	super: fixed typo in <code>\subglossentry</code> (<code>\glossentrydesc</code>)		269
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		143
1.03	<code>\makefirstuc</code> : changed 'protected@edef to 'def		242
1.04	General: Added <code>\glstextformat</code>		81
1.05	<code>\glossarysection</code> : added <code>\mkboth to \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		243
1.06	General: now requires <code>etoolbox</code> ..		241
	<code>\capitalisewords</code> : new		243
	<code>\xcapitalisewords</code> : new		244
1.07	<code>\@gls@link</code> : fixed bug caused by <code>\theglsentrycounter</code> setting the page number too soon ..		94
	<code>\glsadd</code> : fixed bug caused by <code>\theglsentrycounter</code> setting the page number too soon		141
1.08	General: Added babel support ...		30
	<code>\capitalisewords</code> : made robust		243
	<code>listgroup</code> : changed <code>listgroup</code> style to use <code>\glsgetgrouptitle</code>		250
	<code>altlistgroup</code> : changed <code>altlistgroup</code> style to use <code>\glsgetgrouptitle</code>		251
	<code>\makefirstuc</code> : made robust ...		242
1.09	<code>\@mfu@nocaplist</code> : new		244
	<code>\capitalisewords</code> : added check for words that shouldn't be capitalised		243
	<code>\gMFUnocap</code> : new		244
	<code>\mfu@checkword</code> : new		244
	<code>\MFUclear</code> : new		244
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> : numbered sections and auto label added		37
	General: Added support for translator 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> : <code>numberedsection</code> package option added ..		6
	<code>numberline</code> : <code>numberline</code> option added		5

1.12	Removed restriction on only using \newglossaryentry in the preamble	76
\@GLSpl: now uses \glsentrydescplural and \glsentrysymbolplural instead of \glsentrydesc and \glsentrysymbol	110	\newacronym: Removed restriction on only using \newacronym in the preamble 194
\@GLspl@: now uses \glsentrydescplural and \glsentrysymbolplural instead of \glsentrydesc and \glsentrysymbol	109	1.14 \gls@hypergroup: new 246
\@GLspl@: now uses \glsentrydescplural and \glsentrysymbolplural instead of \glsentrydesc and \glsentrysymbol	109	General: added nonumberlist key to \printglossary 180
General: added check for \hypertarget separate to \hyperlink (memoir defines \hyperlink but not \hypertarget)	105	added numberedsection key to \printglossary 178
descriptionplural: new	58	\firstacronymfont: new 198
\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 245
symbolplural support added	70	\glsnavhypertarget: added write to aux file 245
\Glsentrydescplural: New	135	\glsnavigation: changed to only use labels for groups that are present 246
\Glsentrydescplural: New	134	1.15 \gls@link: added \glslabel 94
\Glsentrysymbolplural: New	136	General: Added \glssettoctitle 31
\Glsentrysymbolplural: New	135	\gls@defglossaryentry: check for \@glo@first in description 74
\SetDescriptionFootnoteAcronymStyle: Added \protect before \footnote and \glslink	215	check for \@glo@text in symbol 75
\SetFootnoteAcronymStyle: Added \protect before \footnote and \glslink	222	\gls@hypergroup: new 246
symbolplural: new	59	\glsnavhypertarget: added check if rerun required 245
1.13	General: Add Polish support 350, 353 fixed bug that ignored 3rd parameter 112–120	1.16 \gls@hypergroup: new 246
\ACRfullpl: new	197	\glsnavhypertarget: added check if rerun required 245
\Acrfullpl: new	197	\glssettoctitle: new 29
\acrfullpl: new	196	\printglossary: changed the way the TOC title is set 165
\acrpluralsuffix: New	194	1.16 \@GLS@: Test glossary type is \acronymtype in addition to checking if footnote option has been used 108
\gls@defglossaryentry: Changed default first value	71	\@GLSpl: Test glossary type is \acronymtype in addition to checking if footnote option has been used 110
Changed default firstplural value	71	\@Gls@: Test glossary type is \acronymtype in addition to checking if footnote option has been used 107

\@Glspl@: Test glossary type is \acronymtype in addition to checking if footnote option has been used	110	\glshyperlink:new	140
\@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	191
\@glsdisp: Test glossary type is \acronymtype in addition to checking if footnote option has been used	111	\glshyperlink: replaced 'hy- perlink to '@glslink	245
\@glspl@: Test glossary type is \acronymtype in addition to checking if footnote option has been used	109	\glshypertarget: replaced 'hypertarget to '@glstarget .	245
\@glstarget: raised the hyper- target so the target text doesn't scroll off the top of the page	105	\glssee:new	163
\gls@defglossaryentry: Changed def to let	71	\glsseeformat:new	163
1.17		\glsSetSuffixF:new	34
\@@do@wrglossary:new	160	\glsSetSuffixFF:new	34
\@do@seeglossary:new	162	\ifglsxindy:new	23
\@glo@storeentry:new	77	\istfilename: added xindy sup- port	33
\@glsdefaultplural:new	62	\newglossarystyle: made \newglossarystyle long .	190
\@glsdefaultsort:new	62	\nopostdesc:new	32
\@glshypernumber:new	191	nonumberlist:new	60
\@glsnoname:new	61	\printglossary: added check to determine if \printglossary is already defined	165
\@glsnonextpages:new	181	added print language to aux file	165
\@wrglossary: modified to allow for xindy support	159	order: order package option added	23
General: added Brazilian dictio- nary	354	\writeist: added xindy support	143
Added Brazilian support	350	1.18	
added xindy support	23	\@gls@loadlist:new	8
parent:new	60	\@gls@loadlong:new	8
see:new	60	\@gls@loadsuper:new	8
\gls@defglossaryentry: added nonumberlist key	71	\@gls@loadtree:new	8
added parent key	71	\gls@defglossaryentry: Changed default value of sort to \@glsdefaultsort	71
added see key	71	moved sort sanitization to \newglossaryentry	75
Stored main part of entry format when entry is defined	75	\glstarget:new	184
\gls@glossary: changed defini- tion to use \index instead of @index	158	\oldacronym:new	193
\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	use \textsmaller instead of		first 108
\smaller	219	\@GLSpl: Added check for hyper-	
\SetDescriptionFootnoteAcronymStyle:		first	110
changed \acronymfont to	use \textsmaller instead of	\@GLs@: Added check for hyper-	
\smaller	215	first	107
\SetFootnoteAcronymStyle:		\@GLspl@: Added check for hyper-	
changed \acronymfont to	use \textsmaller instead of	first	110
\smaller	222	\@gls@: Added check for hyper-	
\SetSmallAcronymStyle:		first	106
changed \acronymfont to	use \textsmaller instead of	\@gls@@link: new	93
\smaller	225	\@gls@link: added \leavevmode	
		94
		Moved entry existence check to	
		avoid duplicate code	94
1.2	General: fixed bug in ngerman	\@glsdisp: Added check for hyper-	
	captions	first	111
2.01		\@GLspl@: Added check for hyper-	
	\@gls@link: moved \@do@wrglossary	first	109
	before term is displayed to pre-	\gls@ glossarymark: Added check	
	vent unwanted whatsit	to see if it's already defined ..	36
	95	hyperfirst: new	22
	\forall glossaries: replaced	2.04	
	\ifthenelse with \ifx	\@GLS@: Changed test to check if	
	48	glossary type has been identi-	
	\forall gls entries: replaced	fied as a list of acronyms ...	108
	\ifthenelse with \ifx	\@GLSpl: Changed test to check if	
	48	glossary type has been identi-	
	\glsdefmain: new	fied as a list of acronyms ...	110
	12	\@GLs@: Changed test to check if	
	\glsdescwidth: changed	glossary type has been identi-	
	\linewidth to \hsize . 253,268	fied as a list of acronyms ...	107
	\glslistdottedwidth: changed	\@GLspl@: Changed test to check	
	\linewidth to \hsize	if glossary type has been iden-	
	252	tified as a list of acronyms ..	110
	\gls@pagelistwidth: changed	\@glossaryentryfield: new ..	76
	\linewidth to \hsize . 253,268	\@glossarysubentryfield:	
	nomain: added nomain package	new	76
	option	\@gls@: Changed test to check if	
	13	glossary type has been identi-	
	\writeist: removed item_02 - no	fied as a list of acronyms ...	106
	such makeindex key	\@gls@acronymlists: new	14
	147	\@glsdisp: Changed test to check	
2.02		if glossary type has been iden-	
	\@printglossary: suppressed	tified as a list of acronyms ..	111
	warning globally rather than	\@GLspl@: Changed test to check	
	locally	if glossary type has been iden-	
	167	tified as a list of acronyms ...	109
	General: Changed Brazil to Brazil-		
	ian		
	354		
	false will prevent automatic		
	loading of translator package		
	28		
	\glossarysection: changed		
	\@mkboth to \glossarymark		
	36		
	\gls@ glossarymark: New		
	36		

\@newglossaryentryposthook:	Removed spurious brace. Patch
new 76	provided by Sergiu Dotenco 111
\@newglossaryentryprehook:	\writeist: Added \string be-
new 76	fore opening and closing
acronymlists:new 15	braces. Patch provided by
\DeclareAcronymList:new ... 14	Segiu Dotenco 148
\DefineAcronymSynonyms:new 210	2.06
\gls@defglossaryentry: added	\altnewglossary:new 56
user1-6 keys 71	\CustomAcronymFields:new . 227
\glsadd: fixed bug that ignored	\CustomNewAcronymDef:new . 228
counter 141	\SetCustomDisplayStyle:new 227
\Glsentryuseri:new 137	\SetCustomStyle:new 228
\Glsentryuseri:new 136	2.07
\Glsentryuserii:new 137	General: glssadd format key
\Glsentryuserii:new 137	stored in \@glsnumberformat
\Glsentryuseriii:new 137	(was mistakenly stored in
\Glsentryuseriii:new 137	\@glo@format) 141
\Glsentryuseriv:new 137	3.0
\Glsentryuseriv:new 137	\@@do@wrglossary: added check
\Glsentryuserv:new 137	for hyper location prefix ... 161
\Glsentryuserv:new 137	modified to use new format .. 160
\Glsentryuservi:new 138	\@@glossarysec: replaced
\Glsentryuservi:new 138	\@ifundefined with
\ns@newglossary: added	\ifcsundef 5
check to determine if	\@do@seeglossary: Sanitize and
\gls@<type>@display and	escape cross-referencing in-
\gls@<type>@displayfirst	formation 162
have been defined. 55	\@gls@counterwithin:new 9
\SetAcronymLists:new 15	\@gls@ifinlist:new 39
\SetDefaultAcronymDisplayStyle:	\@gls@link: added \@gls@saveentrycounter
new 212 95
\SetDefaultAcronymStyle:	added \@gls@setsort 95
new 212	\@gls@saveentrycounter:new 95
\SetDescriptionAcronymDisplayStyle:	\@gls@setupsort@def:new ... 11
new 217	\@gls@setupsort@standard:
\SetDescriptionDUAAcronymDisplayStyle:	new 10
new 216	\@gls@setupsort@use:new ... 11
\SetDescriptionFootnoteAcronymDisplayStyle:	\@gls@xdy@locationlist:new 42
new 213	\@gls@link: replaced \@ifundefined
\SetDUADisplayStyle:new .. 225	with \ifcsundef 104
\SetFootnoteAcronymDisplayStyle:	\@glsnextpages:new 181
new 220	\@makeglossary: Added check
\SetSmallAcronymDisplayStyle:	for savewrites 149
new 222	\@print@glossary: replaced
2.05	\@ifundefined with
\@glsdisp: Added closing brace.	\ifcsundef 167
Patch provided by Sergiu	\@printglossary: added
Dotenco 111	\currentglossary 166
	added \@glsnextpages 166

make toctitle default to title ..	166	\GlsAddXdyCounters:new	39
\@set@glo@numformat: added		\glentrycounterlabel:new	183
4th argument	97	\glentryitem:new	183
\@wrglossary: modified to take		\glentrylong:new	138
into account savewrites	159	\glentrylong:new	138
\@xdyattributelist:new	39	\glentrylongpl:new	138
General: added prefix to hyperlink		\glentrylongpl:new	138
.....	192	\glentryshort:new	138
etoolbox now loaded	4	\glentryshort:new	138
replaced \@ifundefined with		\glentryshortpl:new	138
\ifcsundef	28, 91, 178	\glentryshortpl:new	138
\acrfootnote:new	213	\glsgrouptitle: replaced	
\ACRfull: added starred version	196	\@ifundefined with	
\Acrfull: added starred version	196	\ifcsundef	188
\acrfull: added starred version	195	\glsglossarymark: replaced	
\ACRfullpl: added starred ver-		\@ifundefined with	
sion	197	\ifcsundef	36
\Acrfullpl: added starred ver-		\glshyperlink: changed de-	
sion	197	fault from \glentryname to	
\acrfullpl: added starred ver-		\glentrytext	140
sion	196	\glshypernumber: replaced	
\acrlinkfootnote:new	213	\@ifundefined with	
\acrnolinkfootnote:new ...	213	\ifcsundef	191
\addglossarytocaptions: re-		\glnumberformat: replaced	
placed \@ifundefined with		\@ifundefined with	
\ifcsundef	30	\ifcsundef	34
savewrites:new	25	\glhrefentry:new	183
see: added \@glo@seeautonumberlist		\glresetsubentrycounter:	
.....	60	new	182
seeautonumberlist:new	7	\glseeitem: hyperlink uses	
\glossarysection: replaced		\glseeitemformat instead	
\@ifundefined with		of \glentryname	164
\ifcsundef	36	\glseeitemformat:new	164
\glossarystyle: replaced		\glssortnumberfmt:new	10
\@ifundefined with		\glssstepentry:new	182
\ifcsundef	189	\glssstepsubentry:new	182
\glscodepage: replaced		\glssubentrycounterlabel:	
\@ifundefined with		new	183
\ifcsundef	24	\glssubentryitem:new	183
\gl@defglossaryentry: added		theglossary: replaced \@ifundefined	
\@gl@defsort	75	with \ifcsundef	184
added short and long keys	71	short:new	61
replaced \@ifundefined with		shortplural:new	61
\ifcsundef	72	\ifglossaryexists: re-	
\gl@doclearpage: replaced		placed \@ifundefined with	
\@ifundefined with		\ifcsundef	49
\ifcsundef	38	\ifglentryexists: re-	
\gl@add: added \@gl@saveentrycounter		placed \@ifundefined with	
.....	141	\ifcsundef	49

\istfile: deprecated	157	\showglossaryin: new	234
glossaryentry: new	181	\showglossaryout: new	234
glossarysubentry: new	182	\showglossarytitle: new ...	234
\newglossaryentry: replaced		\showglosymbol: new	232
\DeclareRobustCommand		\showglosymbolplural: new .	232
with \newrobustcmd	64	\showglotext: new	230
\newglossarystyle: re-		\showglotype: new	230
placed \@ifundefined with		\showglouserii: new	231
\ifcsundef	190	\showglouseriii: new	231
\ns@newglossary: added		\showglouseriv: new	231
\@gls@defsortcount	55	\showglouserv: new	231
replaced \@ifundefined with		\showglouservi: new	231
\ifcsundef	55	subentrycounter: new	9
entrycounter: new	9	\writeist: added xindy-only	
entrycounterwithin: new	9	macro definitions to glossary	
\oldacronym: replaced \@ifundefined		open tag	145
with \ifcsundef	193	modified to support new for-	
compatible-2.07: compatible-		mat	143
2.07 option added	25		
long: new	61	3.01	
longplural: new	61	\@gls@writefiles: added check	
nonumberlist: now boolean ...	60	for empty glossaries	157
sort: new	10	General: made robust	107
counter: replaced \@ifundefined		\ACRfull: made robust	196
with \ifcsundef	59	\Acrfull: made robust	195
\printglossary: replaced		\acrfull: made robust	195
\@ifundefined with		\acrfullformat: removed	
\ifcsundef	165	\acronymfont as it should al-	
\SetDescriptionFootnoteAcronymDisplayStyle: be set in the second ar-		gument.	195
expanded options link op-		\ACRfullpl: made robust	197
tions	213	\Acrfullpl: made robust	197
\setentrycounter: added op-		\acrfullpl: made robust	196
tional argument	189	\ACRlong: made robust	130
\showacronymlists: new	233	\Acrlong: made robust	129
\showglocounter: new	230	\acrlong: made robust	128
\showglodesc: new	232	\ACRlongpl: made robust	132
\showglodescplural: new ...	232	\Acrlongpl: made robust	131
\showglofirst: new	230	\acrlongpl: made robust	130
\showglofirstpl: new	230	\ACRshort: made robust	126
\showgloflag: new	233	\Acrshort: made robust	126
\showgloindex: new	233	\acrshort: made robust	125
\showglolevel: new	229	\ACRshortpl: made robust	128
\showgloname: new	232	\Acrshortpl: made robust	127
\showgloparent: new	229	\acrshortpl: made robust	127
\showgloplural: new	230	\Gls: made robust	107
\showglosort: new	232	\glsadd: made robust	141
\showglossaries: new	234	\glsaddall: made robust	141
\showglossarycounter: new .	234	\GLSdesc: made robust	117
\showglossaryentries: new .	234		

\Glsdesc: made robust	116	3.02	
\glsdesc: made robust	116		\@do@wrglossary: changed
\GLSdescplural: made robust .	118		\@glslocref to \theglsentrycounter
\Glsdescplural: made robust .	117	 161
\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		159
\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		157
\Glsymbol: made robust	118		\@printglossary: add a way to
\GLSsymbolplural: made robust			fetch current entry label ...
.....	119		167
\Glsymbolplural: made robust			\@wrglossary: added check for
.....	119		glossary file defined
\glsymbolplural: made robust			159
.....	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		164
\glsuseriv: made robust	122		\@glsdisplaynumberlist: new
\GLSuserv: made robust	124		139
\Glsuserv: made robust	124		\@glsentrycounter: set default
\glsuserv: made robust	123		value
\GLSuservi: made robust	125		95
\Glsuservi: made robust	124		\@glsentryfull: fixed bug (re-
\glsuservi: made robust	124		placed \@glsentryshortpl
			with \@glsentryshort)
			139
			\@glsentryfullpl: fixed bug (re-
			placed \@glsentryshort with
			\@glsentryshortpl)
			139
			\@glsentrynumberlist: new ..
			139
			\@glsmoveentry: new
			76
			\@glsnumlistlastsep: new ...
			140
			\@glsnumlistsep: new
			140
			\@glsresetsubentrycounter:
			new
			182
			\@ifglshaschildren: new
			50
			\@ifglshasparent: new
			51
			\@makeglossaries: added list
			parser
			152
			indexonlyfirst: new
			22
			\@renewglossarystyle: new ..
			190

\showglossaryentries: fixed misspelt command	234	super3col: added check for glsnogroupskip	271
\SmallNewAcronymDef: fixed broken short and long plural	223	super4col: added check for glsnogroupskip	272
3.03		superragged: added check for glsnogroupskip	276
\@gls@sanitizesort:new	18	superragged3col: added check for glsnogroupskip	278
\@gls@setupsort@standard: used \@gls@sanitizesort .	10	3.04	
\@printglossary: allow title to override default toctitle	166	\@do@wrglossary: changed \theglsentrycounter back to \@glslocref	161
General: allow title to set toctitle	178	\@do@wrglossary: modified to compensate for possible incorrect page number	160
\glsinlinedescformat:new .	249	\@gls@escbsdq: unsanitize \gls@numberpage, \gls@alphpage, \gls@Alphpage and \gls@romanpage	98
\glsinlineemptydescformat:new	249	\@print@glossary: Moved aux write to end of document to prevent unwanted whatsit occurring here.	167
\glsinlinenameformat:new .	249	General: Added check for doc package	4
\glsinlinepostchild:new ..	249	added datatool-base as a required package	4
\glsinlinesubdescformat:new	249	added local key	92
\glsinlinesubnameformat:new	249	\gls@Alphpage:new	160
\glspostinline: replaced “.” with \glspostdescription	249	\gls@alphpage:new	159
altlongragged4col: added check for glsnogroupskip ..	263	\gls@disablepagerefexpansion:new	159
altsuperragged4col: added check for glsnogroupskip ..	279	\gls@numberpage:new	160
alttree: added check for glsnogroupskip	287	\gls@protected@pagefmts:new	159
index: added check for glsnogroupskip	282	\gls@romanpage:new	160
nogroupskip:new	9	\glsdefmain: added check for doc package	12
long: added check for glsnogroupskip	253	\glsorg@endtheglossary:new .	5
long3col: added check for glsnogroupskip	255	\glsorg@glossary:new	4
long4col: added check for glsnogroupskip	256	\glsorg@theglossary:new	5
longragged: added check for glsnogroupskip	260	\glsorg@wrglossary:new	4
longragged3col: added check for glsnogroupskip	261	altlist: replaced \newline with paragraph break	251
nopostdot:new	9	\PrintChanges:new	5
tree: added check for glsnogroupskip	283	3.05	
treenoname: added check for glsnogroupskip	285	\@do@wrglossary: add Roman case. Fixed bugs in the else statements	160
super: added check for glsnogroupskip	269		

\@gls@link: added check for “no-hypertypes”	94	General: added nogroupskip key to \printglossary	179
\@gls@nohyperlist: new	16	removed definition of \@glossaryentryfield ..	328
mcolalttree: replaced ‘2’ with \glsmcols	267	removed definition of \@glossarysubentryfield	328
mcolindex: replaced ‘2’ with \glsmcols	265	\compatibleglossentry: new	184
mcoltree: replaced ‘2’ with \glsmcols	265	\compatiblesubglossentry: new	186
mcoltreenoname: replaced ‘2’ with \glsmcols	266	\glossaryentryfield: deprecated	186
\gls@protected@pagefmts: added Roman to list	159	\Glossentrydesc: new	185
\gls@Romanpage: new	160	\glossentrydesc: new	185
\GlsDeclareNoHyperList: new	16	\Glossentryname: new	185
\glsgetgrouplabel: fixed bug (typo in \equal)	189	\glossentryname: new	185
\nopostdesc: made robust	32	\Glossentrysymbol: new	185
nohypertypes: new	16	\glossentrysymbol: new	185
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	141
\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	163
\GlsSetXdyCodePage: Added check for fontspec	47	\glsseeformat: made robust ..	163
\SetDescriptionAcronymDisplayStyle: now using \glsdoparenifnotempty	217	\glsseeitem: made robust	164
\setglossarypreamble: new ..	35	\glsseelist: made robust	163
3.08a		\ifglshasdescsuppressed: 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	250
\@glossarysubentryfield: switched to \subglossentry	76	listdotted: updated listdotted style to use \glossentry and \subglossentry	252
		altlist: updated altlist style to use \glossentry and \subglossentry	251
		altlongragged4col: updated to use \glossentry and \subglossentry	262
		alttree: updated to use \glossentry and \subglossentry	286
		index: added paragraph break at end of environment	281

updated to use \glossentry and \subglossentry	281	\Glsentryuser: made robust .	137
inline: updated inline style to use \glossentry and \subglossentry	248	\Glsentryuseri: made robust	138
long: updated to use \glossentry and \subglossentry	253	\glstextup: new	194
longragged: updated to use \glossentry and \subglossentry	259	\if@gl@docloaded: Add a fix for \RecordChanges	4
longragged3col: updated to use \glossentry and \subglossentry	261	\ifglshassymbol: changed test to check for \@gl@default@symbol	51
tree: updated to use \glossentry and \subglossentry	283	3.10a	
\setglossarystyle: new	189	\@gl@keymap: new	66
\setglossentrycompatibility: new	186	\@gl@provide@newglossary: new	53
superragged: updated to use \glossentry and \subglossentry	276	\@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	168
\Glsentrydesc: made robust ..	134	\gl@assign@type@field: new	17
\Glsentrydescplural: made ro- bust	135	\gl@defglossaryentry: Changed to using \@gl@default@value	71
\Glsentryfirst: made robust .	136	new	70
\Glsentryfirstplural: made robust	136	\glswritedefhook: new	69
\Glsentryfull: made robust ..	139	\makeglossaries: Added providecommand code to aux file	151
\Glsentryfullpl: made robust	139	\new@glossaryentry: new	65
\Glsentrylong: made robust ..	138	\ns@newglossary: added \@gl@provide@newglossary	55
\Glsentrylongpl: made robust	138	3.11a	
\Glsentryname: made robust ..	133	\@ACRlong: added \glslabel, \gl@ifplural, \glscapscase, \gl\$insert and \glscustomtext	327
\Glsentryplural: made robust	135	\@ACRshort: added \glslabel, \gl@ifplural, \glscapscase, \gl\$insert and \glscustomtext	326
\Glsentryshort: made robust .	138	\@Acrlong: added \glslabel, \gl@ifplural, \glscapscase, \gl\$insert and \glscustomtext	327
\Glsentryshortpl: made robust	138	\@Acrshort: added \glslabel, \gl@ifplural, \glscapscase, \gl\$insert and \glscustomtext	326
\Glsentrysymbol: made robust	135		
\Glsentrysymbolplural: made robust	136		
\Glsentrytext: made robust ..	135		
\Glsentryuseri: made robust .	137		
\Glsentryuserii: made robust	137		
\Glsentryuseriii: made robust	137		
\Glsentryuseriv: made robust	137		

<p>\@GLS@: add \glslabel, \glsifplural,\glscapscase, \glscustomtext and \glsinsert 108 change to using \glsentryfmt style commands 108 removed \MakeUppercase (now moved to \glsentryfmt) 108 \@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 \@GLs@: add \glsifplural, \glscapscase,\glscustomtext and \glsinsert 107 change to using \glsentryfmt style commands 107 removed \makefirstuc (now dealt with in \glsentryfmt) 107 \@GLspl@: add \glsifplural, \glscapscase,\glscustomtext and \glsinsert 109 change to using \glsentryfmt style commands 109 removed \makefirstuc (now dealt with in \glsentryfmt) 110 \@acrlong: added \glslabel, \glsifplural,\glscapscase, \glsinsert and \glscustomtext 327 \@acrshort: added \glslabel, \glsifplural,\glscapscase, \glsinsert and \glscustomtext 326 \@gls@: add \glslabel, \glsifplural,\glscapscase, \glscustomtext and \glsinsert 106 change to using \glsentryfmt style commands 106 \@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 \@glspl@: add \glslabel, \glsifplural,\glscapscase, \glscustomtext and \glsinsert 109 change to using \glsentryfmt style commands 109 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 247
..... 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 227
..... 124, 125	\SetDefaultAcronymDisplayStyle:
changed to just use \Glsentryuserv	changed to use \defglssentryfmt
..... 124 212
changed to just use \glssentryserv	\SetDescriptionAcronymDisplayStyle:
..... 123, 124	updated to use \defglssentryfmt
Now requires textcase 3 217
acronymlists: replaced	\SetDescriptionDUAAcronymDisplayStyle:
\@addtoacronymlists with	updated to use \defglssentryfmt
\DeclareAcronymList 15 216
\defglssdisplay: obsoleted 90	\SetDescriptionFootnoteAcronymDisplayStyle:
\defglssdisplayfirst: obso-	updated to use \defglssentryfmt
leted 90 213
\defglssentryfmt: new 54	\SetDUADisplayStyle: updated
\forglssentries: replaced \ifx	to use \defglssentryfmt .. 225
with \ifdefempty 48	\SetFootnoteAcronymDisplayStyle:
\gls@assign@desc: new 69	updated to use \defglssentryfmt
\gls@defglossaryentry: Fixed 220
default counter if none sup-	\SetSmallAcronymDisplayStyle:
plied 74	updated to use \defglssentryfmt
\gls@doentryfmt: new 54 222
\glssdisplay: obsoleted 90	\setupglossaries: new 27
\glssdisplayfirst: obsoleted .. 90	\showglolong: new 233
\glssgenentryfmt: new 85	\showgloshort: new 233
\glssgetgrouptitle: Added	numbers: new 26
check in case non-Latin alpha-	symbols: new 25
bet in use 188	

3.12a		\glsentryfullpl: changed to use \acrfullformat 139
\gls@defglossaryentry: added		
\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 274	
..... 18	altsuper4colheaderborder:	
\@gls@link: removed \relax .. 95	switched to \tabularnewline	
\@gls@notranslatorhook: new 21 274	
\@gls@setupsort@standard:	long: switched to \tabularnewline	
moved \@gls@santizesort 253	
to \glsprestandardsort .. 10	long3col: switched to \tabularnewline	
General: added cs@gls@notranslatorhook 255	
to else clause 32	long3colheader: switched to	
ucmark: added check for memoir . 9	\tabularnewline 255	
see: added \gls@checkseeallowed	long3colheaderborder: switched	
..... 60	to \tabularnewline 256	
\glossarysection: changed	long4col: switched to \tabularnewline	
\glossarymark to \gls glossarymark 256	
..... 36	long4colheader: switched to	
\glossarystyle: fixed bug	\tabularnewline 257	
caused by using \ifdef in-	longheader: switched to	
stead of \ifcsdef 189	\tabularnewline 254	
\gls@assign@desc@field:	longheaderborder: switched to	
changed to use \glssetnoexpandfield	\tabularnewline 254	
..... 17	\SetFootnoteAcronymDisplayStyle:	
\gls@assign@descplural@field:	fixed missing argument bug 220	
changed to use \glssetnoexpandfieldsuper:	switched to \tabularnewline	
..... 17 269	
\gls@assign@name@field:	super3col: switched to	
changed to use \glssetnoexpandfield	\tabularnewline 270	
..... 17	super3colheader: switched to	
\gls@assign@type@field:	\tabularnewline 271	
changed to use \glssetexpandfield	super4col: switched to	
..... 17	\tabularnewline 272	
\gls@checkseeallowed: new .. 60	super4colheader: switched to	
\glsaddallunused: set default to	\tabularnewline 272	
\@glo@types 141	super4colheaderborder:	
\Glsentryfull: changed to use	switched to \tabularnewline	
\acrfullformat 139 273	
\glsentryfull: changed to use	superheader: switched to	
\acrfullformat 139	\tabularnewline 269	
\Glsentryfullpl: changed to	superheaderborder: switched to	
use \acrfullformat 139	\tabularnewline 270	

3.14a		\genacrfullformat:new 89
\@glswritefiles:	renamed	\GenericAcronymFields:new 199
\glswritefiles to \@glswritefiles	and used “savewrites” option	\Genplacrfullformat:new . . . 90
to set \glswritefiles 157		\genplacrfullformat:new . . . 89
General:new 235		\Glsentryfull: bug fix: added
acronyms:new 14		missing \acronymfont 139
\gls@defglossaryentry: added		\Glsentryfull: bug fix: added
check for existence of default		missing \acronymfont 139
glossary 71		\Glsentryfullpl: bug fix: added
set the default for firstplural to		missing \acronymfont 139
be the value of plural 73		\Glsentryfullpl: bug fix: added
xindygloss:new 24		missing \acronymfont 139
\longprovideglossaryentry:		\glsgenacfmt:new 87
new 70		\GlsUseAcrEntryDisplayStyle:
compatible-2.07: added check		new 201
for 2.07 before setting 3.07		\GlsUseAcrStyleDefs:new . . 201
compatibility 25		short-long:new 202
notranslate:new 21		short-long-desc:new 204
\provideglossaryentry:new . 64		xindynoglsnumbers:new 24
4.0		sm-short-long:new 203
\gls@defglossaryentry: added		sm-short-long-desc:new . . . 205
check for first key 73		\makeglossaries: made pream-
4.01		ble only 152
General: fixed non-value options		index:new 26
so that they can be passed to		\newacronymstyle:new 200
document class 7		long-sc-short:new 202
\CustomAcronymFields: in-		long-sc-short-desc:new . . . 203
serted missing comma 228		long-short:new 201
4.02		long-short-desc:new 203
\@acrfull: now using \acrfullfmt		long-sm-short:new 202
. 195		long-sm-short-desc:new . . . 204
\@gls@indexdef:new 26		footnote:new 207
\@gls@numbersdef:new 26		footnote-desc:new 209
\@gls@symbolsdef:new 26		footnote-sc:new 208
General: Removed \acronymfont		footnote-sc-desc:new 209
. 129–132		footnote-sm:new 209
\ACRfullfmt:new 196		footnote-sm-desc:new 209
\Acrfullfmt:new 196		\setacronymstyle:new 200
\acrfullfmt:new 195		\SetDescriptionAcronymDisplayStyle:
\ACRfullplfmt:new 197		Moved check for empty cus-
\Acrfullplfmt:new 197		tom text to prevent unwanted
\acrfullplfmt:new 197		parenthetical material 217
\acronymentry:new 199		\SetDescriptionFootnoteAcronymDisplayStyle:
sanitize: fixed bug that caused		Moved check for empty cus-
an error here 21		tom text to prevent unwanted
sc-short-long:new 203		parenthetical material 214
sc-short-long-desc:new . . . 204		\SetFootnoteAcronymDisplayStyle:
\Genacrfullformat:new 89		Moved check for empty cus-

tom text to prevent unwanted parenthetical material	220	\@glsdisp: removed \glslabel (defined in \@gls@link) . . .	111
\SetGenericNewAcronym: new	198	\@glspl@: removed \glslabel (defined in \@gls@link) . . .	109
\SetSmallAcronymDisplayStyle: Moved check for empty cus- tom text to prevent unwanted parenthetical material	222	\@printglossary: added \glsdetoklabel	167
dua: new	205	General: changed default to \@empty instead of \relax . .	25
dua-desc: new	207	removed \glslabel (defined in \@gls@link)	125
numberedsection: added		sc-short-long-desc: redefined to use accessibility informa- tion	332
nameref option	6	\compatibleglossentry: added \glsdetoklabel	308
4.03		\compatiblesubglossentry: added \glsdetoklabel . . .	309
\@do@wrglossary: added		\Genacrfullformat: redefined to use accessibility informa- tion	325
\glsdetoklabel	161	\genacrfullformat: redefined to use accessibility informa- tion	325
\@ACRlong: removed \glslabel (defined in \@gls@link) . . .	327	\Genplacrfullformat: rede- fined to use accessibility in- formation	325
\@ACRshort: removed \glslabel (defined in \@gls@link) . . .	326	\genplacrfullformat: rede- fined to use accessibility in- formation	325
\@Acrlong: removed \glslabel (defined in \@gls@link) . . .	327	\glossentryname: added \glsdetoklabel	185
\@ACRshort: removed \glslabel (defined in \@gls@link) . . .	326	\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	141
\@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) . . .	327	\glsdisplaynumberlist: added \glsdetoklabel	140
\@acrshort: removed \glslabel (defined in \@gls@link) . . .	326	\glsdoifexistsorwarn: new . .	50
\@gls@: removed \glslabel (de- fined in \@gls@link)	106	\glsentryaccess: switched to using \@gls@entry@field .	313
\@gls@access@display: new .	314	\glsentrydescaccess: switched to using \@gls@entry@field	313
\@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> 313	<code>\glsstepentry:added\glsdetoklabel</code> 182
<code>\glsentryfirstaccess:switched</code> to using <code>\@gls@entry@field</code> 313	<code>\glsstepsubentry: added</code> <code>\glsdetoklabel</code> 182
<code>\glsentryfirstplural: added</code> <code>\glsdetoklabel</code> 136	<code>\glsunset:added\glsdetoklabel</code> 79
<code>\glsentrylongaccess: switched</code> to using <code>\@gls@entry@field</code> 314	short-long: commented spuri- ous EOL 202
<code>\glsentrylongpluralaccess:</code> switched to using <code>\@gls@entry@field</code> 314	redefined to use accessibility in- formation 330
<code>\glsentrypluralaccess:</code> switched to using <code>\@gls@entry@field</code> 313	short-long-desc: redefined to use accessibility information 332
<code>\glsentryshortaccess:switched</code> to using <code>\@gls@entry@field</code> 314	<code>\ifglsdessuppressed: added</code> <code>\glsdetoklabel</code> 51
<code>\glsentryshortpluralaccess:</code> switched to using <code>\@gls@entry@field</code> 314	fixed typo 51
<code>\glsentrysymbolaccess:</code> switched to using <code>\@gls@entry@field</code> 313	<code>\ifglshaschildren: added</code> <code>\glsdetoklabel</code> 49
<code>\glsentrysymbolpluralaccess:</code> switched to using <code>\@gls@entry@field</code> 313	<code>\ifglshasdesc:added\glsdetoklabel</code> 51
<code>\glsentrytextaccess: switched</code> to using <code>\@gls@entry@field</code> 313	<code>\ifglshasfield:new</code> 52
<code>\glsgenacfmt: redefined to use</code> accessibility information ... 323	<code>\ifglshaslong:added\glsdetoklabel</code> 51
<code>\glsgenentryfmt: redefined to</code> use accessibility information 320	<code>\ifglshasparent: added</code> <code>\glsdetoklabel</code> 51
<code>\glshyperlink:added\glsdetoklabel</code> 140	<code>\ifglshasshort: added</code> <code>\glsdetoklabel</code> 52
<code>\glslocalreset: added</code> <code>\glsdetoklabel</code> 79	<code>\ifglshassymbol: added</code> <code>\glsdetoklabel</code> 51
<code>\glslocalunset: added</code> <code>\glsdetoklabel</code> 79	replaced <code>\ifcseempty</code> with <code>\ifdefempty</code> and replaced <code>\ifx</code> with <code>\ifdefequal</code> 51
<code>\glsmoveentry:added\glsdetoklabel</code> 76	<code>\ifglused:added\glsdetoklabel</code> 49
replaced <code>\ifthenelse</code> with <code>\ifdefequal</code> 76	sm-short-long-desc: redefined to use accessibility informa- tion 332
<code>\glsrefentry:added\glsdetoklabel</code> 183	long-sc-short-desc: redefined to use accessibility informa- tion 331
<code>\glsreset:added\glsdetoklabel</code> 78	long-short: redefined to use ac- cessibility information 329
<code>\glsseelist:added\expandafter</code> commands 164	long-short-desc: redefined to use accessibility information 331
	long-sm-short-desc: redefined to use accessibility informa- tion 331

footnote: redefined to use accessibility information	335	\showgloplural: added	
footnote-desc: redefined to use accessibility information . . .	337	\glsdetoklabel	230
footnote-sc: redefined to use accessibility information	337	\showglopluralaccess: added	
footnote-sc-desc: redefined to use accessibility information	338	\glsdetoklabel	344
footnote-sm: redefined to use accessibility information	337	\showgloshort: added \glsdetoklabel	
footnote-sm-desc: redefined to use accessibility information	338	233
\renewacronymstyle: new . . .	200	\showgloshortaccess: added	
\showglocounter: added		\glsdetoklabel	344
\glsdetoklabel	230	\showgloshortpluralaccess:	
\showgloDESC: added \glsdetoklabel		added \glsdetoklabel . . .	344
.....	232	\showglosort: added \glsdetoklabel	
\showgloDESCaccess: added		232
\glsdetoklabel	344	\showglosymbol: added	
\showgloDESCplural: added		\glsdetoklabel	232
\glsdetoklabel	232	\showglosymbolaccess: added	
\showgloDESCpluralaccess:		\glsdetoklabel	344
added \glsdetoklabel . . .	344	\showglosymbolplural: added	
\showgloFIRST: added \glsdetoklabel		\glsdetoklabel	232
.....	230	\showglosymbolpluralaccess:	
\showgloFIRSTaccess: added		added \glsdetoklabel . . .	344
\glsdetoklabel	344	\showglotext: added \glsdetoklabel	
\showgloFIRSTpl: added		230
\glsdetoklabel	230	\showglotextaccess: added	
\showgloFIRSTpluralaccess:		\glsdetoklabel	344
added \glsdetoklabel . . .	344	\showglotype: added \glsdetoklabel	
\showgloFLAG: added \glsdetoklabel		230
.....	233	\showglouserI: added \glsdetoklabel	
\showgloINDEX: added \glsdetoklabel		231
.....	233	\showglouserII: added	
\showgloLEVEL: added \glsdetoklabel		\glsdetoklabel	231
.....	229	\showglouserIII: added	
\showgloLONG: added \glsdetoklabel		\glsdetoklabel	231
.....	233	\showglouserIV: added	
\showgloLONGaccess: added		\glsdetoklabel	231
\glsdetoklabel	345	\showglouserV: added \glsdetoklabel	
\showgloLONGpluralaccess:		231
added \glsdetoklabel . . .	345	\showglouserVI: added	
\showgloNAME: added \glsdetoklabel		\glsdetoklabel	231
.....	232	dua: fixed bug in \acrfullfmt .	206
\showgloNAMEaccess: added		fixed bug in \Acrfullplfmt .	206
\glsdetoklabel	343	fixed bug in \acrfullplfmt .	206
\showgloPARENT: added		redefined to use accessibility in-	
\glsdetoklabel	229	formation	332
		dua-desc: commented spurious	
		EOL	207
		redefined to use accessibility in-	
		formation	335

4.04		
\@gls@noidx@nosanitizesort:		\def as is may or may not be
new	18	defined
\@gls@noidx@sanitizesort:		308
new	18	\compatiblesubglossentry:
\@gls@nosanitizesort:new ..	18	changed \newcommand to
\@gls@sanitizesort:new ...	18	\def as is may or may not be
\@glo@addchildren:new	169	defined
\@glo@do@sortentries:new ..	170	309
\@glo@grabfirst:new	175	\defglsdisplayfirst: fixed un-
\@glo@sortedinsert:new ...	170	wanted space
\@glo@sortentries:new	169	91
\@glo@sorthandler@case:new	171	\glo@grabfirst:new
\@glo@sorthandler@letter:		174
new	171	\gls@defglossaryentry: re-
\@glo@sorthandler@nocase:		placed \ifx with \ifdefvoid
new	171	75
\@glo@sorthandler@word:new	171	\glsnoidxdisplayloc:new ..
\@glo@sortmacro@case:new ..	172	177
\@glo@sortmacro@def:new ..	173	\glsnoidxdisplaylocclisthandler:
\@glo@sortmacro@def@do:new	173	new
\@glo@sortmacro@letter:new	172	177
\@glo@sortmacro@nocase:new	173	\glsnoidxloclist:new
\@glo@sortmacro@standard:		176
new	172	\glsnoidxlocclisthandler:
\@glo@sortmacro@use:new ..	174	new
\@glo@sortmacro@word:new ..	171	177
\@gls@getcounterprefix:		\glsnoidxstripaccents:new ..
added warning if no prefix can		19
be formed	162	alttree: moved hangindent and
\@gls@getothergrouptitle:		parindent assignments out-
new	188	side level test
\@gls@noidx@do:new	175	286
\@gls@noref@warn:new	156	\makeglossaries: Moved def-
\@gls@reference:new	177	inition of \glswrite to
\@gls@warnonglossdefined:		\makeglossaries
new	16	151
\@gls@warnontheGLOSSdefined:		\makenoidxglossaries:new ..
new	16	153
\@no@makeglossaries:new ..	156	\printglossary: changed to use
\@print@glossary:new	167	new \@printglossary ...
\@print@noidx@glossary:new	174	165
\@printgloss@setsort:new ..	166	\printnoidxglossaries:new
\@printglossary:new	166	165
General: added sort key to print-		\@printnoidxglossary:new ..
gloss group	180	165
\compatibleglossentry:		\showgloclist:new
changed \newcommand to		233
		\warn@noprintglossary: Acti-
		vate warning in \makeglossaries
	
		165
		\writeist: checked for definition
		of \glswrite
		143, 147
	4.06	
		\@GLS@: added \glsifhyper ..
		108
		\@GLSpl: added \glsifhyper ..
		110
		\@Gls@: added \glsifhyper ..
		107
		\@Glspl@: added \glsifhyper
		109
		\@gls@: added \glsifhyper ..
		106
		\@gls@numbersdef: added hook
		to set toc title
		26
		\@gls@symbolsdef: added hook
		to set toc title
		26
		\@glsdisp: added \glsifhyper
		111
		\@glspl@: added \glsifhyper
		109
		General: added \glsifhyper ..
	
		126–132

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 \@glstype	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	175	\@glsdisp: moved \@glsifhyper	111
\Acrfullplfmt: fixed no case change bug	197	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	327	\@ignored@glossaries: new ..	57
\@ACRshort: added \do@gls@link@checkfirsthyper	326	General: added entrycounter option to printgloss family ..	179
\@Acrlong: added \do@gls@link@checkfirsthyper	327	added \nopostdot option to printgloss family	179
\@Acrshort: added \do@gls@link@checkfirsthyper	326	added \subentrycounter option to printgloss family	180
\@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	327	removed \@sACRshortpl ...	128
\@acrshort: added \do@gls@link@checkfirsthyper	326	removed \@sAcrshortpl ...	127
\@closegls: new	150	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	238
removed \@sGlsname	116	\ACRfull: removed \s@ACRfull	196
removed \@sglsname	115	switched to using \@gls@hyp@opt	
removed \@sGLSplural	114	196
removed \@sGlsplural	114	\Acrfull: removed \s@Acrfull	196
removed \@sglsplural	113	switched to using \@gls@hyp@opt	
removed \@sGLSpl	110	195
removed \@sGlspl	109	\acrfull: removed \s@acrfull	195
removed \@sglspl	108	switched to using \@gls@hyp@opt	
removed \@sGLSsymbolplural		195
.....	119	\ACRfullpl: removed \s@ACRfullpl	
removed \@sGLssymbolplural		197
.....	119	switched to using \@gls@hyp@opt	
removed \@sglssymbolplural		197
.....	119	\Acrfullpl: removed \s@Acrfullpl	
removed \@sGLSsymbol	119	197
removed \@sGLssymbol	118	switched to using \@gls@hyp@opt	
removed \@sglssymbol	118	197
removed \@sGLStext	112	\acrfullpl: removed \s@acrfullpl	
removed \@sGlstext	112	196
removed \@sglstext	112	switched to using \@gls@hyp@opt	
removed \@sGLSuseriii ...	122	196
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	240	\acrshortpl: switched to using	
removed sPgls	239	\@gls@hyp@opt	127
removed spgls	238	\forallacronyms: new	48
removed sPGLSpl	241	\GLS: switched to using	
removed sPglspl	239	\@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 195
\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 281
\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	\newglossary: added starred ver- sion	54
\Glsuserii: switched to using \@gls@hyp@opt	121	\newignoredglossary: new ...	57
\glsuserii: switched to using \@gls@hyp@opt	121	\ns@newglossary: added \@glotype@<name>@log ...	55
\GLSuseriii: switched to using \@gls@hyp@opt	122	new	55
\Glsuseriii: switched to using \@gls@hyp@opt	122	\p@gls@hyp@opt: new	93
\glsuseriii: switched to using \@gls@hyp@opt	121	\PGLS: changed to use \@gls@hyp@opt	240
\GLSuseriv: switched to using \@gls@hyp@opt	123	\Pgls: changed to use \@gls@hyp@opt	239
\Glsuseriv: switched to using \@gls@hyp@opt	123	\pgls: changed to use \@gls@hyp@opt	238
\glsuseriv: switched to using \@gls@hyp@opt	122	\PGLSpl: changed to use \@gls@hyp@opt	241
\GLSuserv: switched to using \@gls@hyp@opt	124	\Pglspl: changed to use \@gls@hyp@opt	239
\Glsuserv: switched to using \@gls@hyp@opt	124	\pglspl: changed to use \@gls@hyp@opt	238
\glsuserv: switched to using \@gls@hyp@opt	123	\s@gls@hyp@opt: new	92
\GLSuservi: switched to using \@gls@hyp@opt	125	\s@newglossary: new	54
\Glsuservi: switched to using \@gls@hyp@opt	124	automake: new	24
\glsuservi: switched to using \@gls@hyp@opt	124		
\ifignoredglossary: new	57	4.09	
altlongragged4col: fixed bug that displayed description in- stead of symbol	262	\glsaddkey: fixed bug in user commands	68
		4.10	
		\@Gls@acentryname: new ...	133
		\@Gls@entryname: new	133
		\gls@glossary: Renamed \@glossary to \@gls@glossary	158
		\glspercentchar: new	142
		\glstildechar: new	142
		almtree: moved space after sym- bol	286, 287

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	161
\@do@wrglossary	160
\@glo@assign@sortkey	181
\@glossarysec	5
\@glossaryseclabel	6
\@glossarysecstar	6
\@gls@default@entryfmt	316
\@gls@expand@field	63
\@gls@fixbraces	163
\@gls@noidx@nosanitizesort .	18
\@gls@noidx@sanitizesort ...	18
\@gls@nosanitizesort	18
\@gls@sanitizesort	18
\@ACRlong	327
\@ACRshort	326
\@Acrlong	327
\@Acrshort	326
\@GLS@	108
\@GLSpl	110
\@Gls@	107
\@Gls@acrentryname	133
\@Gls@entry@field	133
\@Gls@entryname	133
\@Glspl@	109
\@PGLS	240
\@PGLS@	240
\@PGLSpl	241
\@PGLSpl@	241
\@Pgls	239
\@Pgls@	239
\@Pglspl	239
\@Pglspl@	240
\@acrfull	195
\@acrlong	327
\@acrshort	326
\@addtoacronymlists	14
\@closegls	150
\@delimN	192
\@delimR	191
\@disable@onlypremakeg	29
\@disable@premakecs	29
\@disabled@glsaddxdycounters	40
\@do@seeglossary	162
\@do@wrglossary	159, 290
\@glo@addchildren	169
\@glo@default@sorttype	9
\@glo@do@sortentries	170
\@glo@grabfirst	175
\@glo@no@assign@sortkey	181
\@glo@seeautonumberlist	7
\@glo@sortedinsert	170
\@glo@sortentries	169
\@glo@sorthandler@case	171
\@glo@sorthandler@letter ...	171
\@glo@sorthandler@nocase ...	171
\@glo@sorthandler@word	171
\@glo@sortmacro@case	172
\@glo@sortmacro@def	173
\@glo@sortmacro@def@do	173
\@glo@sortmacro@letter	172
\@glo@sortmacro@nocase	173
\@glo@sortmacro@standard ...	172
\@glo@sortmacro@use	174
\@glo@sortmacro@word	171
\@glo@storeentry	76
\@glo@types	53
\@glossary@default@style	6
\@glossaryentryfield	76
\@glossarysection	37
\@glossarysubentryfield	76
\@gls	106
\@gls@	106
\@gls@@link	93
\@gls@access@display	314
\@gls@addpredefinedattributes	
.....	41
\@gls@assign@symbol@field ...	18
\@gls@assign@symbolplural@field	
.....	18
\@gls@checkactual	102
\@gls@checkbar	101
\@gls@checkescactual	100
\@gls@checkescbar	100
\@gls@checkesclevel	101
\@gls@checkescquote	99

<code>\@gls@checklevel</code>	102	<code>\@gls@sanitizesort</code>	18
<code>\@gls@checkmkidxchars</code>	98	<code>\@gls@sanitizesymbol</code>	18
<code>\@gls@checkquote</code>	99	<code>\@gls@saveentrycounter</code>	95
<code>\@gls@codepage</code>	47	<code>\@gls@setacrstyle</code>	22
<code>\@gls@counterwithin</code>	9	<code>\@gls@setcounter</code>	56
<code>\@gls@declareoption</code>	7	<code>\@gls@setupsort@def</code>	11
<code>\@gls@default@value</code>	59	<code>\@gls@setupsort@standard</code>	10
<code>\@gls@do@acronymsdef</code>	13	<code>\@gls@setupsort@use</code>	11
<code>\@gls@doautomake</code>	25	<code>\@gls@startswithexpandonce</code>	64
<code>\@gls@entry@field</code>	132	<code>\@gls@symbolsdef</code>	26
<code>\@gls@escbsdq</code>	97	<code>\@gls@tmpb</code>	99
<code>\@gls@expand@fields</code>	63	<code>\@gls@toc</code>	38
<code>\@gls@fetchfield</code>	66	<code>\@gls@updatechecked</code>	99
<code>\@gls@field@link</code>	111	<code>\@gls@warnonglossdefined</code>	16
<code>\@gls@fixbraces</code>	163	<code>\@gls@warnontheGLOSSdefined</code>	16
<code>\@gls@forbidtexext</code>	54	<code>\@gls@writedef</code>	65
<code>\@gls@getcounter</code>	56	<code>\@gls@xdy@Lclass@Alpha-page-numbers</code>	43
<code>\@gls@getcounterprefix</code>	162	<code>\@gls@xdy@Lclass@Appendix-page-numbers</code>	43
<code>\@gls@getgrouptitle</code>	188	<code>\@gls@xdy@Lclass@Roman-page-numbers</code>	43
<code>\@gls@getothergrouptitle</code>	188	<code>\@gls@xdy@Lclass@alpha-page-numbers</code>	43
<code>\@gls@glossary</code>	158	<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>	246	<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, 236	<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>	142
<code>\@gls@makefirsttuc</code>	243	<code>\@gls@hypernumber</code>	191
<code>\@gls@missingnumberlist</code>	62	<code>\@gls@link</code>	104
<code>\@gls@noaccess</code>	311	<code>\@gls@minrange</code>	143
<code>\@gls@noexpand@field</code>	62	<code>\@gls@nextpages</code>	181
<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>	175	<code>\@gls@nonextpages</code>	181
<code>\@gls@noidx@setsanitizesort</code>	20	<code>\@gls@openfile</code>	149
<code>\@gls@noref@warn</code>	156	<code>\@gls@order</code>	23
<code>\@gls@notranslatorhook</code>	21	<code>\@gls@spl@</code>	108
<code>\@gls@numbersdef</code>	26		
<code>\@gls@onlypremakeg</code>	28		
<code>\@gls@provide@newglossary</code>	53		
<code>\@gls@reference</code>	177		
<code>\@gls@renewglossary</code>	158		
<code>\@gls@sanitizedesc</code>	17		
<code>\@gls@sanitizename</code>	17		

short-long 202, 330
 short-long-desc 204, 332
 sm-short-long 203
 sm-short-long-desc .. 205, 332
 \acronymentry 199
 \acronymfont
 87, 198, 215, 219, 222, 225
 acronymlists (option) 15
 \acronymname 29
 acronyms (option) 14
 \acronymsort 200
 \acronymtype 13, 194
 \acrpluralsuffix 194
 \ACRshort 126
 \Acrshort 126
 \acrshort 125
 \ACRshortpl 128
 \Acrshortpl 127
 \acrshortpl 127
 \Acs 210
 \acs 210
 \Acsp 210
 \acsp 210
 \addglossarytocaptions 30
 \addto 30
 align (environment) 78, 95, 96
 altlist (style) 251
 altlistgroup (style) 251
 altlisthypergroup (style) 251
 altlong4col (style) 257
 altlong4colborder (style) 258
 altlong4colheader (style) 258
 altlong4colheaderborder (style) 258
 altlongragged4col (style) 262
 altlongragged4colborder (style) 263
 altlongragged4colheader (style) 263
 altlongragged4colheaderborder
 (style) 264
 \altnewglossary 56
 altsuper4col (style) 273
 altsuper4colborder (style) 274
 altsuper4colheader (style) 274
 altsuper4colheaderborder
 (style) 274
 altsuperragged4col (style) 279
 altsuperragged4colborder
 (style) 280
 altsuperragged4colheader
 (style) 279

altsuperragged4colheaderborder
 (style) 280
 alttree (style) 285
 alttreegroup (style) 288
 alttreehypergroup (style) 288
 amsgen package 4, 91
 amsmath package 78
 \andname 30
 array package 259, 275
 article class 162
 automake (option) 24

B

babel package 28, 29, 31, 46, 345

C

\capitalisewords 243
 \changes 5
 \compatglossarystyle 295
 compatible-2.07 (option) 25
 compatible-3.07 (option) 25
 \compatibleglossentry .. 184, 308
 \compatiblesubglossentry 186, 309
 counter (key) 59
 counter (option) 15
 \CustomAcronymFields 227
 \CustomNewAcronymDef 228

D

datatool package 170
 \DeclareAcronymList 14
 \DefaultNewAcronymDef .. 212, 338
 \defentryfmt 91
 \defglstdisplay 90
 \defglstdisplayfirst 90
 \defglssentry 55
 \defglssentryfmt 54, 58, 59, 82
 \DefineAcronymSynonyms 210
 \delimN 35, 191
 \delimR 35, 191
 description (environment) 249, 250
 description (key) 58
 description (option) 22
 descriptionaccess (key) 310
 \DescriptionDUANewAcronymDef 216
 \DescriptionFootnoteNewAcronymDef
 214, 340
 \descriptionname 30
 \DescriptionNewAcronymDef ..
 218, 340

descriptionplural (key) 58
descriptionpluralaccess (key) 310
\detokenize 49
doc package 4, 5, 12
dua (acrstyle) 205, 332
dua (option) 23
dua-desc (acrstyle) 207, 335
\DUANewAcronymDef 225

E

entrycounter (option) 9
entrycounterwithin (option) 9
\entryname 29
environments:
 align 78, 95, 96
 description 249, 250
 longtable 8, 229, 252–264
 multicols 264
 supertabular ... 8, 229, 268–280
 theglossary 5,
 16, 35, 184, 190, 266, 267, 283–286
 theindex 281
equation (counter) 95, 96
etoolbox package 4, 241

F

file types
 .aux 167
 .glo 77
 .ist 142, 149
 .toc 38
 .xdy 33
 glo 235
\findrootlanguage 46
first (key) 59
firstaccess (key) 310
\firstacronymfont 87, 198
firstplural (key) 59
firstpluralaccess (key) 310
footnote (acrstyle) 207, 335
footnote (option) 22
footnote-desc (acrstyle) .. 209, 337
footnote-sc (acrstyle) 208, 337
footnote-sc-desc (acrstyle) 209, 338
footnote-sm (acrstyle) 209, 337
footnote-sm-desc (acrstyle) 209, 338
\FootnoteNewAcronymDef . 220, 341
\forallacronyms 48
\forallglossaries 48
\forallglsentries 48

\forallglsentries 48

G

garamondx package 194
\Genacrfullformat 89, 325
\genacrfullformat 89, 325
\GenericAcronymFields 199
\Genplacrfullformat 90, 325
\genplacrfullformat 89, 325
\glo@grabfirst 174
\glolinkprefix 95
glossareentry (counter) 182
glossaries package 26,
 46, 47, 143, 229, 235, 250, 288, 308
glossaries-accsupp package ... 76, 308
\GlossariesWarning 16
\GlossariesWarningNoLine 16
\glossary 54, 149, 189
glossary counters:
 glossaryentry 181
 glossarysubentry 182
glossary keys:
 access 309
 counter 59
 description 58
 descriptionaccess 310
 descriptionplural 58
 descriptionpluralaccess . 310
 first 59
 firstaccess 310
 firstplural 59
 firstpluralaccess 310
 long 61
 longaccess 310
 longplural 61
 longpluralaccess 311
 name 58
 nonumberlist 60
 parent 60
 plural 58
 pluralaccess 310
 see 60
 short 61
 shortaccess 310
 shortplural 61
 shortpluralaccess 310
 sort 58
 symbol 59
 symbolaccess 310

symbolplural	59	altsuperragged4col	279, 280, 306
symbolpluralaccess	310	altsuperragged4col	279
text	58	altsuperragged4colborder	
textaccess	309	280, 306
type	59	altsuperragged4colborder	280
user1	60	altsuperragged4colheader	
user2	60	279, 306
user3	61	altsuperragged4colheader	279
user4	61	altsuperragged4colheaderborder	
user5	61	280, 306
user6	61	altsuperragged4colheaderborder	
glossary package	1, 193	280
glossary styles:		alttree	267, 285, 288, 302
altlist	251, 296	alttree	285
altlist	251	alttreegroup	288, 304
altlistgroup	251, 297	alttreegroup	288
altlistgroup	251	alttreehypergroup	288, 304
altlisthypergroup	251, 297	alttreehypergroup	288
altlisthypergroup	251	index	264, 281, 282, 300
altlong4col	257, 258, 262, 299	index	281
altlong4col	257	indexgroup	282, 301
altlong4colborder	258, 299	indexgroup	282
altlong4colborder	258	indexhypergroup	282, 301
altlong4colheader	258, 299	indexhypergroup	282
altlong4colheader	258	inline	295
altlong4colheaderborder		inline	247
.....	258, 299	list	6, 249–252, 296
altlong4colheaderborder	258	list	249
altlongragged4col	262, 263, 300	listdotted	252, 297
altlongragged4col	262	listdotted	252
altlongragged4colborder		listgroup	250, 296
.....	263, 300	listgroup	250
altlongragged4colborder	263	listhypergroup	250, 296
altlongragged4colheader		listhypergroup	250
.....	263, 300	long	253, 254, 259, 297, 299
altlongragged4colheader	263	long	253
altlongragged4colheaderborder		long3col	254, 255, 298
.....	264, 300	long3col	254
altlongragged4colheaderborder		long3colborder	255, 298
.....	264	long3colborder	255
altsuper4col	273, 274, 279, 308	long3colheader	255, 298
altsuper4col	273	long3colheader	255
altsuper4colborder	274, 308	long3colheaderborder	255, 298
altsuper4colborder	274	long3colheaderborder	255
altsuper4colheader	274, 308	long4col	256, 257, 298
altsuper4colheader	274	long4col	256
altsuper4colheaderborder		long4colborder	257, 298
.....	274, 308	long4colborder	257
altsuper4colheaderborder	274	long4colheader	256, 298

long4colheader	256	mcoltreenonamehypergroup	
long4colheaderborder	257, 298	266, 304
long4colheaderborder	mcoltreenonamehypergroup	266
longborder	254, 297	sublistdotted	297
longborder	254	sublistdotted	252
longheader	254, 297	super	268–270, 276, 306
longheader	254	super	268
longheaderborder	super3col	270, 271, 307
longheaderborder	super3col	270
longragged	259–261	super3colborder
longragged	259	super3colborder
longragged3col ...	261, 262, 299	super3colborder
longragged3col	261	super3colheader
longragged3colborder	261, 300	super3colheader
longragged3colborder	super3colheader
longragged3colheader	262, 300	super3colheaderborder	271, 307
longragged3colheader	super3colheaderborder	... 271
longragged3colheaderborder		super4col	272, 273, 307
.....	262, 300	super4col	272
longragged3colheaderborder	super4colborder
longraggedborder	super4colborder
longraggedborder	super4colborder
longraggedheader	super4colheader
longraggedheader	super4colheader
longraggedheaderborder	260, 299	super4colheaderborder	273, 308
longraggedheaderborder	.. 260	super4colheaderborder	... 273
mcolalttree	267, 305	superragged	269, 306
mcolalttree	267	superragged	269
mcolalttreegroup	superragged	269, 307
mcolalttreegroup	superragged	269
mcolalttreehypergroup	267, 305	superragged3col ..	277, 278, 305
mcolalttreehypergroup	... 267	superragged3col	277
mcolindex	265, 304	superragged3colborder	278, 306
mcolindex	264	superragged3colborder	... 278
mcolindexgroup	superragged3colheader	278, 306
mcolindexgroup	superragged3colheader	... 278
mcolindexhypergroup	. 265, 304	superragged3colheaderborder	
mcolindexhypergroup	278, 306
mcoltree	265, 304	superraggedborder	... 276, 305
mcoltree	265	superraggedborder
mcoltreegroup	superraggedborder
mcoltreegroup	superraggedheader	... 276, 305
mcoltreegroup	superraggedheader
mcoltreehypergroup	.. 266, 304	superraggedheaderborder	.
mcoltreehypergroup	277, 305
mcoltreenoname	superraggedheaderborder	. 277
mcoltreenoname	superraggedright3colheaderborder	
mcoltreenonamegroup	. 266, 304	278
mcoltreenonamegroup	tree	265, 282–285, 301

tree	282	\gls@Alphpage	160
treegroup	266, 283, 302	\gls@alphpage	159
treegroup	283	\gls@assign@desc	69
treehypergroup	283, 302	\gls@assign@desc@field	17
treehypergroup	283	\gls@assign@descplural@field	17
treenoname ...	266, 284, 285, 302	\gls@assign@field	64
treenoname	284	\gls@assign@name@field	17
treenonamegroup	285, 302	\gls@assign@type@field	17
treenonamegroup	285	\gls@checkisacronymlist	15
treenonamehypergroup	285, 302	\gls@checkseeallowed	60
treenonamehypergroup	285	\gls@codepage	24
glossary-hypernav package	142	\gls@defglossaryentry	70
glossary-list package	6, 8, 249	\gls@disablepagerefexpansion	159
glossary-long package		\gls@docclearpage	38
.....	8, 252, 253, 262, 268	\gls@doentryfmt	54
glossary-longragged package	259	\gls@glossary	158
glossary-mcols package	264	\gls@hypergroupprerun	246
glossary-super package		\gls@ifnotmeasuring	78
.....	8, 253, 268, 275, 279	\gls@istfilebase	33
glossary-superragged package	275	\gls@level	62
glossary-tree package	8, 281	\gls@noidxglossary	157
glossaryentry (counter) .	9, 182, 183	\gls@numberpage	160
glossaryentry (counter)	181	\gls@protected@pagefmts	159
\glossaryentryfield	186, 190	\gls@Romanpage	160
\glossaryentrynumber	181	\gls@romanpage	160
\glossaryentrynumbers		\gls@save@numberlist	164
.....	7, 34, 166, 167	\gls@suffixF	34
\glossaryheader	184, 190	\gls@suffixFF	34
\glossarymark	36	\gls@accessdisplay	316
\glossaryname	29, 30	\gls@accsupp	314
\glossarypostamble	36, 190	\glsadd	80, 140, 141, 189
\glossarypreamble	35, 190	\glsadd options	
\glossarysection	6, 36, 54	counter	141
\glossarystyle	189, 229	format	141, 190
glossarysubentry (counter) ...		\glsaddall	49, 80, 141
.....	9, 182, 183	\glsaddall options	
glossarysubentry (counter) ...	182	types	141
\glossarysubentryfield	186	\glsaddallunused	141
\glossentry	59, 184	\glsaddkey	67
\Glossentrydesc	185	\GlsAddLetterGroup	47
\glossentrydesc	185, 328	\GlsAddSortRule	45
\Glossentryname	185	\GlsAddXdyAlphabet	42
\glossentryname	185, 328	\GlsAddXdyAttribute	40, 288
\Glossentrysymbol	185	\GlsAddXdyCounters	39, 289
\glossentrysymbol	185, 328	\GlsAddXdyLocation	44, 289
\GLS	107	\GlsAddXdyStyle	46
\Gls	107, 109, 242	\glsautoprefix	6
\gls	4, 59,	\gls@clearpage	38
	80, 92, 106–108, 112–124, 183, 237	\gls@closebrace	142

<code>\glsc compositor</code>	33, 44	<code>\Glsentrylong</code>	138
<code>\glscounter</code>	15, 56	<code>\glscounterlong</code>	138
<code>\GlsDeclareNoHyperList</code>	16	<code>\glscounterlongaccess</code>	314
<code>\glscdefaultttype</code>	13	<code>\Glsentrylongpl</code>	138
<code>\glscdefmain</code>	12	<code>\glscounterlongpl</code>	138
<code>\GLSdesc</code>	117	<code>\glscounterlongpluralaccess</code>	314
<code>\Glsdesc</code>	116	<code>\Glsentryname</code>	133
<code>\glscdesc</code>	116, 117	<code>\glscentryname</code>	133, 164
<code>\GLSdescplural</code>	118	<code>\glscentrynumberlist</code>	139, 154
<code>\Glsdescplural</code>	117	<code>\Glsentryplural</code>	135
<code>\glscdescplural</code>	117, 118	<code>\glscentryplural</code>	135
<code>\glscdescriptionaccessdisplay</code>	315	<code>\glscentrypluralaccess</code>	313
<code>\glscdescriptionpluralaccessdisplay</code>	315	<code>\Glsentryprefix</code>	237
<code>\glscdescwidth</code>	253, 259, 268, 275	<code>\glscentryprefix</code>	236
<code>\glscdetoklabel</code>	49	<code>\Glsentryprefixfirst</code>	236
<code>\glscdisablehyper</code>	105	<code>\glscentryprefixfirst</code>	236
<code>\glscdisp</code>	111	<code>\Glsentryprefixfirstplural</code>	237
<code>\glscdisplay</code>	81, 90, 106	<code>\glscentryprefixfirstplural</code>	236
<code>\glscdisplayfirst</code>	81, 90, 106	<code>\Glsentryprefixplural</code>	237
<code>\glscdisplaynumberlist</code>	139, 155, 177	<code>\glscentryprefixplural</code>	236
<code>\glscdohyperlink</code>	104	<code>\Glsentryshort</code>	138
<code>\glscdohypertarget</code>	104	<code>\glscentryshort</code>	138
<code>\glscdoifexists</code>	50	<code>\glscentryshortaccess</code>	314
<code>\glscdoifexistsorwarn</code>	50	<code>\Glsentryshortpl</code>	138
<code>\glscdoifnoexists</code>	50	<code>\glscentryshortpl</code>	138
<code>\glscdoparenifnotempty</code>	222	<code>\glscentryshortpluralaccess</code>	314
<code>\glscenablehyper</code>	105	<code>\glscentrysort</code>	136
<code>\glscentryaccess</code>	313	<code>\Glsentrysymbol</code>	135
<code>\glscentrycounter</code>	95	<code>\glscentrysymbol</code>	135
<code>\glscentrycounterlabel</code>	183	<code>\glscentrysymbolaccess</code>	313
<code>\Glsentrydesc</code>	134	<code>\Glsentrysymbolplural</code>	136
<code>\glscentrydesc</code>	134	<code>\glscentrysymbolplural</code>	135
<code>\glscentrydescaccess</code>	313	<code>\glscentrysymbolpluralaccess</code>	313
<code>\Glsentrydescplural</code>	135	<code>\Glsentrytext</code>	135
<code>\glscentrydescplural</code>	134	<code>\glscentrytext</code>	49, 135, 164
<code>\glscentrydescpluralaccess</code>	313	<code>\glscentrytextaccess</code>	313
<code>\Glsentryfirst</code>	136	<code>\glscentrytype</code>	136
<code>\glscentryfirst</code>	136	<code>\Glsentryuseri</code>	137
<code>\glscentryfirstaccess</code>	313	<code>\glscentryuseri</code>	136
<code>\Glsentryfirstplural</code>	136	<code>\Glsentryuserii</code>	137
<code>\glscentryfirstplural</code>	136	<code>\glscentryuserii</code>	137
<code>\glscentryfirstpluralaccess</code>	313	<code>\Glsentryuseriii</code>	137
<code>\glscentryfmt</code>	58, 59, 81	<code>\glscentryuseriii</code>	137
<code>\Glsentryfull</code>	139	<code>\Glsentryuseriv</code>	137
<code>\glscentryfull</code>	139, 199, 208, 337	<code>\glscentryuseriv</code>	137
<code>\Glsentryfullpl</code>	139	<code>\Glsentryuserv</code>	137
<code>\glscentryfullpl</code>	139	<code>\glscentryuserv</code>	137
<code>\glscentryitem</code>	183	<code>\Glsentryuservi</code>	138
		<code>\glscentryuservi</code>	138

<code>\glsexpandfields</code>	64	<code>\glslongaccessdisplay</code>	316
<code>\GLSfirst</code>	113	<code>\glslongaccesskey</code>	343
<code>\Glsfirst</code>	113	<code>\glslongkey</code>	195
<code>\glsfirst</code>	112, 113	<code>\glslongpluralaccessdisplay</code>	316
<code>\glsfirstaccessdisplay</code>	315	<code>\glslongpluralaccesskey</code>	343
<code>\GLSfirstplural</code>	115	<code>\glslongpluralkey</code>	195
<code>\Glsfirstplural</code>	115	<code>\glslongtok</code>	198
<code>\glsfirstplural</code>	114, 115	<code>\glsmakefirsttuc</code>	243
<code>\glsfirstpluralaccessdisplay</code>	315	<code>\glsmccls</code>	264
<code>\glsngenacfmt</code>	87, 323	<code>\glsmoveentry</code>	76
<code>\glsngenentryfmt</code>	85, 320	<code>\GLSname</code>	116
<code>\glsgetgrouplabel</code>	189	<code>\Glsname</code>	115
<code>\glsgetgrouptitle</code>	142, 188	<code>\glsname</code>	115, 116
<code>\gls glossarymark</code>	9, 36	<code>\glsnameaccessdisplay</code>	314
<code>\gls groupheading</code>	187, 190	<code>\glsnamefont</code>	190, 281
<code>\gls groupskip</code>	187, 190, 249	<code>\glsnavhyperlink</code>	245
<code>\glshyperlink</code>	140	<code>\glsnavhypertarget</code>	245
<code>\glshypernavsep</code>	247	<code>\glsnavigation</code>	246
<code>\glshypernumber</code>	34, 191	<code>\glsnextpages</code>	181
<code>\glsifhyper</code>	92	<code>\glsnoexpandfields</code>	64
<code>\glsifhyperon</code>	92	<code>\glsnoidxdisplayloc</code>	177
<code>\glsIfListOfAcronyms</code>	14	<code>\glsnoidxdisplaylocclsthandler</code>	177
<code>\glsinlinedescformat</code>	249	<code>\glsnoidxlocclst</code>	176
<code>\glsinlinedopostchild</code>	247, 248	<code>\glsnoidxlocclsthandler</code>	177
<code>\glsinlineemptydescformat</code>	249	<code>\glsnoidxnumberlistloophandler</code>	156
<code>\glsinlinenameformat</code>	249	<code>\glsnoidxstripaccents</code>	19
<code>\glsinlineparentchildseparator</code>	249	<code>\glsnonextpages</code>	181
<code>\glsinlinepostchild</code>	249	<code>\glsnoxindywarning</code>	39
<code>\glsinlineseparator</code>	249	<code>\glsnumberformat</code>	34
<code>\glsinlinesubdescformat</code>	249	<code>\glsnumberlistloop</code>	156
<code>\glsinlinesubnameformat</code>	249	<code>\glsnumbersgroupname</code>	30, 142, 187
<code>\glsinlinesubseparator</code>	249	<code>\glsnumlistlastsep</code>	140
<code>\glskeylisttok</code>	198	<code>\glsnumlistsep</code>	140
<code>\glslabeltok</code>	198	<code>\glsopenbrace</code>	142
<code>\glsletentryfield</code>	132	<code>\glsorder</code>	23
<code>\glslink</code>	80, 81, 93, 106, 140, 189, 190	<code>\glsorg@endtheglossary</code>	5
<code>\glslink options</code>		<code>\glsorg@glossary</code>	4
counter	91, 106, 235	<code>\glsorg@theglossary</code>	5
format	91, 106, 190	<code>\glsorg@wrglossary</code>	4
hyper	92, 93, 106	<code>\glspagelistwidth</code>	253, 259, 268, 275
local	92	<code>\glspar</code>	32
<code>\glslinkcheckfirsthyperhook</code>	94	<code>\glsperscentchar</code>	142
<code>\glslinkvar</code>	92	<code>\GLSpl</code>	110
<code>\glslistdottedwidth</code>	252	<code>\Glspl</code>	109, 242
<code>\glslocalreset</code>	79	<code>\glspl</code>	80, 108–110
<code>\glslocalresetall</code>	79	<code>\GLSplplural</code>	114
<code>\glslocalunset</code>	79	<code>\Glsplural</code>	114
<code>\glslocalunsetall</code>	80		

<code>\glsplural</code>	113, 114	<code>\glssymbol</code>	118
<code>\glspluralaccessdisplay</code>	315	<code>\glssymbolaccessdisplay</code>	315
<code>\glspluralsuffix</code>	30, 58, 59	<code>\glssymbolnav</code>	247
<code>\glspostdescription</code>	8	<code>\GLSsymbolplural</code>	119
<code>\glspostinline</code>	249	<code>\Glsymbolplural</code>	119
<code>\glsprestandardsort</code>	10	<code>\glssymbolplural</code>	119
<code>\glsquote</code>	142	<code>\glssymbolpluralaccessdisplay</code>	
<code>\glsrefentry</code>	183	315
<code>\glsreset</code>	78	<code>\glssymbolsgroupname</code> .	30, 142, 187
<code>\glsresetall</code>	79	<code>\glstarget</code>	184
<code>\glsresetentrylist</code>	181	<code>\GLStext</code>	112
<code>\glsresetsubentrycounter</code> ...	182	<code>\Glstext</code>	112
<code>\glssee</code>	163	<code>\glstext</code>	112
<code>\glsseeformat</code>	145, 163	<code>\glstextaccessdisplay</code>	315
<code>\glsseeitem</code>	164	<code>\glstextformat</code>	81
<code>\glsseeitemformat</code>	164	<code>\glstextup</code>	194
<code>\glsseelastsep</code>	164	<code>\glstildechar</code>	142
<code>\glsseelist</code>	163	<code>\glstreeindent</code>	283, 284
<code>\glsseesep</code>	164	<code>\glstreenamefmt</code>	281
<code>\glsSetAlphaCompositor</code>	34	<code>\glsunset</code>	79
<code>\glsSetCompositor</code>	33	<code>\glsunsetall</code>	80
<code>\glssetexpandfield</code>	17	<code>\GlsUseAcrEntryDispStyle</code> ...	201
<code>\glssetnoexpandfield</code>	17	<code>\GlsUseAcrStyleDefs</code>	201
<code>\glsSetSuffixF</code>	34	<code>\GLSuseri</code>	120
<code>\glsSetSuffixFF</code>	34	<code>\Glsuseri</code>	120
<code>\glssettoctitle</code>	29	<code>\glsuseri</code>	120
<code>\glssetwidest</code>	285	<code>\GLSuserii</code>	121
<code>\GlsSetXdyCodePage</code>	47	<code>\Glsuserii</code>	121
<code>\GlsSetXdyFirstLetterAfterDigits</code>		<code>\glsuserii</code>	121
.....	143	<code>\GLSuseriii</code>	122
<code>\GlsSetXdyLanguage</code>	46	<code>\Glsuseriii</code>	122
<code>\GlsSetXdyLocationClassOrder</code>	45	<code>\glsuseriii</code>	121, 122
<code>\GlsSetXdyMinRangeLength</code> ...	143	<code>\GLSuseriv</code>	123
<code>\GlsSetXdyStyles</code>	46	<code>\Glsuseriv</code>	123
<code>\glsshortaccessdisplay</code>	315	<code>\glsuseriv</code>	122, 123
<code>\glsshortaccesskey</code>	343	<code>\GLSuserv</code>	124
<code>\glsshortkey</code>	194	<code>\Glsuserv</code>	124
<code>\glsshortpluralaccessdisplay</code>	315	<code>\glsuserv</code>	123, 124
<code>\glsshortpluralaccesskey</code> ...	343	<code>\GLSuservi</code>	125
<code>\glsshortpluralkey</code>	195	<code>\Glsuservi</code>	124
<code>\glsshorttok</code>	198	<code>\glsuservi</code>	124, 125
<code>\glssortnumberfmt</code>	10	<code>\glswrite</code>	153
<code>\glsspace</code>	195	<code>\glswritedefhook</code>	69
<code>\glsstepentry</code>	182	<code>\gmFUnocap</code>	244
<code>\glsstepsubentry</code>	182		
<code>\glssubentrycounterlabel</code> ...	183		
<code>\glssubentryitem</code>	183		
<code>\GLSsymbol</code>	118		
<code>\Glsymbol</code>	118		

H			
<code>\hyperbf</code>	193		
<code>\hyperemph</code>	193		
<code>hyperfirst (option)</code>	22		

`\hyperit` 193
`\hyperlink` 104
`\hypermd` 193
`\hyperpage` 191
`hyperref` package ... 162, 165, 191, 235
`\hyperm` 192
`\hypersc` 193
`\hypersf` 192
`\hypersl` 193
`\hypertarget` 105
`\hypertt` 192
`\hyperup` 193

I

`\if@gls@docloaded` 4
`\if@gls@isacronymlist` 15
`\ifglossaryexists` 49
`\ifgl$descsuppressed` 51
`\ifgl$entryexists` 49
`\ifgl$haschildren` 50
`\ifgl$hasdesc` 51
`\ifgl$hasfield` 52
`\ifgl$haslong` 51
`\ifgl$hasparent` 51
`\ifgl$hasprefix` 237
`\ifgl$hasprefixfirst` 237
`\ifgl$hasprefixfirstplural` . 237
`\ifgl$hasprefixplural` 237
`\ifgl$hasshort` 52
`\ifgl$hasymbol` 51
`\ifgl$translate` 21
`\ifgl$used` 49, 78
`\ifgl$xyndy` 23
`\ifignoredglossary` 57
`index` (option) 26
`index` (style) 281
`indexgroup` (style) 282
`indexhypergroup` (style) 282
`indexonlyfirst` (option) 22
`inline` (style) 247
`\inputencodingname` 24
`\istfile` 157
`\istfilename` 33
`\item` 190, 249, 281

L

`link` text 81
`list` (style) 249
`listdotted` (style) 252
`listgroup` (style) 250

`listhypergroup` (style) 250
`\loadglsentries` 12, 80
`long` (key) 61
`long` (style) 253
`long-sc-short` (acrstyle) 202
`long-sc-short-desc` (acrstyle) ..
..... 203, 331
`long-short` (acrstyle) 201, 329
`long-short-desc` (acrstyle) . 203, 331
`long-sm-short` (acrstyle) 202
`long-sm-short-desc` (acrstyle) ..
..... 204, 331
`long3col` (style) 254
`long3colborder` (style) 255
`long3colheader` (style) 255
`long3colheaderborder` (style) .. 255
`long4col` (style) 256
`long4colborder` (style) 257
`long4colheader` (style) 256
`long4colheaderborder` (style) .. 257
`longaccess` (key) 310
`longborder` (style) 254
`longheader` (style) 254
`longheaderborder` (style) 254
`\longnewglossaryentry` 58, 69
`longplural` (key) 61
`longpluralaccess` (key) 311
`\longprovideglossaryentry` ... 70
`longragged` (style) 259
`longragged3col` (style) 261
`longragged3colborder` (style) .. 261
`longragged3colheader` (style) .. 262
`longragged3colheaderborder`
(style) 262
`longraggedborder` (style) 260
`longraggedheader` (style) 260
`longraggedheaderborder` (style) 260
`longtable` (environment)
..... 8, 229, 252–264
`longtable` package 253, 259

M

`\makefirstuc` 241
`makeglossaries`
..... 23, 33, 46, 47, 54, 151, 167
`\makeglossaries`
... 24, 28, 32–34, 53, 56, 151, 153
`\makeglossary` 153
`makeindex` 358

makeindex 10, 23, 24, 30, 33–35, 54,
 56, 58, 77, 97, 100, 142, 145,
 147, 149, 158, 161, 162, 187, 290
 delim_n 34
 delim_r 35
 page_compositor 33
 special characters 98, 99, 142
 makeindex (option) 23
 \makenoidxglossaries 20, 153
 mcolalttree (style) 267
 mcolalttreegroup (style) 267
 mcolalttreehypergroup (style) . 267
 mcolindex (style) 264
 mcolindexgroup (style) 265
 mcolindexhypergroup (style) ... 265
 mcoltree (style) 265
 mcoltreegroup (style) 265
 mcoltreehypergroup (style) 266
 mcoltreenoname (style) 266
 mcoltreenonamegroup (style) ... 266
 mcoltreenonamehypergroup
 (style) 266
 memoir class 158
 mfirstuc package 1, 244
 \mfirstucMakeUppercase 243
 \mfu@checkword 244
 \MFUclear 244
 \MFUocap 244
 multicol package 264
 multicol (environment) 264

N

name (key) 58
 \new@glossaryentry 65
 \newacronym 22, 23, 61, 80, 81, 193, 194
 \newacronymhook 198
 \newacronymstyle 200
 \newglossary 15, 54, 56, 149, 151, 178
 \newglossaryentry 58, 64, 80, 81, 194
 \newglossaryentry options
 access 311, 313
 counter 59
 description 22, 57, 58,
 62, 64, 70, 116, 134, 194, 221, 310
 descriptionaccess 313, 315
 descriptionplural 117, 310
 descriptionpluralaccess .. 313, 315
 first 59, 73,
 105, 112, 136, 219, 224, 225, 310

firstaccess 313, 315
 firstplural 59, 114, 136, 310
 firstpluralaccess 313, 315
 format 144
 long 87, 138, 310
 longaccess 314, 316
 longplural 138, 311
 longpluralaccess 314, 316
 name 57,
 58, 61, 64, 70, 115, 133, 164, 309
 nonnumberlist 60
 parent 60, 64
 plural 58, 73, 113, 310
 pluralaccess 313, 315
 prefix 236
 prefixfirst 236
 prefixfirstplural 236
 prefixplural 236
 see 7, 60, 152, 153
 short 87, 138, 310
 shortaccess 314, 315
 shortplural 138, 310
 shortpluralaccess 314, 315
 sort 58, 136, 187
 symbol 58, 59, 118, 215,
 217, 219, 224, 256, 272, 309–311
 symbolaccess 313, 315
 symbolplural 119, 310
 symbolpluralaccess 313, 315
 text 58,
 59, 105, 112, 135, 215, 219, 309
 textaccess 313, 315
 type 12, 59, 80, 136
 user1 120, 136, 311
 user2 121, 137
 user3 121, 137
 user4 122, 137
 user5 123, 137
 user6 124, 138, 311
 \newglossarystyle 190
 \newignoredglossary 57
 nogroupskip (option) 9
 nohypertypes (option) 16
 \noist 149, 235, 295
 nolist (option) 8
 nolong (option) 8
 nomain (option) 13
 nonnumberlist (key) 60
 nonnumberlist (option) 7

<code>\nopostdesc</code>	32
<code>nopostdot</code> (option)	9
<code>noredefwarn</code> (option)	16
<code>nostyles</code> (option)	8
<code>nosuper</code> (option)	8
<code>notranslate</code> (option)	21
<code>notree</code> (option)	8
<code>nowarn</code> (option)	16
<code>\ns@newglossary</code>	55
<code>numberedsection</code> (option)	6
<code>numberline</code> (option)	5
<code>numbers</code> (option)	26

O

<code>\oldacronym</code>	193
<code>order</code> (option)	23

P

<code>\p@glshyp@opt</code>	93
package options:	
<code>acronym</code>	13, 29, 165, 194
<code>true</code>	14
<code>acronym</code>	13
<code>acronymlists</code>	15
<code>acronyms</code>	14
<code>automake</code>	24
<code>compatible-2.07</code>	25
<code>compatible-3.07</code>	25
<code>counter</code>	15
<code>counter</code>	15
<code>description</code>	219, 220
<code>description</code>	22
<code>dua</code>	217, 219, 220
<code>dua</code>	23
<code>entrycounter</code>	180, 181
<code>true</code>	9
<code>entrycounter</code>	9
<code>entrycounterwithin</code>	9
<code>footnote</code> 106–111, 215, 217, 219, 221	
<code>footnote</code>	22
<code>hyperfirst</code>	
<code>false</code>	106–111
<code>hyperfirst</code>	22
<code>index</code>	26
<code>indexonlyfirst</code>	365
<code>indexonlyfirst</code>	22
<code>makeindex</code>	145, 235
<code>makeindex</code>	23
<code>nogroupskip</code>	9
<code>nohypertypes</code>	16

<code>nolist</code>	229
<code>nolist</code>	8
<code>nolong</code>	229, 253
<code>nolong</code>	8
<code>nomain</code>	12, 13
<code>nomain</code>	13
<code>nonumberlist</code>	7
<code>nonumberlist</code>	7
<code>nopostdot</code>	9
<code>noredefwarn</code>	16
<code>nostyles</code>	8
<code>nosuper</code>	229
<code>nosuper</code>	8
<code>notranslate</code>	21
<code>notree</code>	229
<code>notree</code>	8
<code>nowarn</code>	16
<code>numberedsection</code>	6
<code>numberline</code>	5
<code>numberline</code>	5
<code>numbers</code>	26
<code>order</code>	23
<code>sanitize</code>	19, 58, 133, 134
<code>sanitize</code>	21
<code>sanitizesort</code>	20
<code>savenumberlist</code>	7
<code>savewrites</code>	25, 362, 363
<code>false</code>	149
<code>true</code>	151, 157
<code>savewrites</code>	25
<code>section</code>	6, 37
<code>section</code>	6
<code>seeautonumberlist</code>	7
<code>shotcuts</code>	23
<code>smallcaps</code>	23
<code>smaller</code>	23
<code>sort</code>	
<code>def</code>	10
<code>standard</code>	10
<code>use</code>	10
<code>sort</code>	10
<code>style</code>	7, 229
<code>style</code>	7
<code>subentrycounter</code>	180, 182
<code>subentrycounter</code>	9
<code>symbols</code>	25
<code>toc</code>	5
<code>true</code>	5
<code>toc</code>	5

translate	21
false	21
translate	21
translator	21
ucmark	9
xindy	24, 145, 235
xindy	24
xindygloss	24
xindynoglsnumbers	24
\pagelistname	30
parent (key)	60
\PGLS	240
\Pgls	239
\pgls	238
\PGLSpl	241
\Pglspl	239
\pglspl	238
\phantomsection	36, 37
plural (key)	58
pluralaccess (key)	310
polyglossia package	28, 30
\printacronyms	13
\PrintChanges	5
\printglossaries	
.. 12, 35, 53, 56, 57, 153, 165, 245	
\printglossary	
35, 54, 57, 153, 165, 167, 178, 245	
\printglossary options	
entrycounter	179
nogroupskip	179
nonumberlist	180
nopostdot	179
numberedsection	178
style	178
subentrycounter	180
title	178
toctitle	178
type	12, 164, 178
\printnoidxglossaries	165
\printnoidxglossary	165, 178
\printnoidxglossary options	
sort	180
\provideglossaryentry	64
 R	
\renewacronymstyle	200
\renewglossarystyle	190
\roman	43

S

\s@glshyp@opt	92
\s@newglossary	54
sanitize (option)	21
sanitizesort (option)	20
savenumberlist (option)	7
savewrites (option)	25
sc-short-long (acrstyle)	203
sc-short-long-desc (acrstyle) ..	
..... 204, 332	
\scantokens	49
\section	49
section (option)	6
see (key)	60
seeautonumberlist (option)	7
\seename	30
\SetAcronymLists	15
\SetAcronymStyle	14, 227
\setacronymstyle	200
\SetCustomDisplayStyle	227
\SetCustomStyle	228
\SetDefaultAcronymDisplayStyle	
..... 212	
\SetDefaultAcronymStyle	212
\SetDescriptionAcronymDisplayStyle	
..... 217	
\SetDescriptionAcronymStyle	219
\SetDescriptionDUAAcronymDisplayStyle	
..... 216	
\SetDescriptionDUAAcronymStyle	
..... 217	
\SetDescriptionFootnoteAcronymDisplayStyle	
..... 213	
\SetDescriptionFootnoteAcronymStyle	
..... 215	
\SetDUADisplayStyle	225
\SetDUASyle	226
\setentrycounter	189
\SetFootnoteAcronymDisplayStyle	
..... 220	
\SetFootnoteAcronymStyle ...	221
\SetGenericNewAcronym	198
\setglossarypreamble	35
\setglossarysection	37
\setglossarystyle	189
\setglossentrycompatibility	186
\SetSmallAcronymDisplayStyle	222
\SetSmallAcronymStyle	224
\setStyleFile	32, 33

<code>\setupglossaries</code>	27	<code>\showglouseriii</code>	231
<code>short (key)</code>	61	<code>\showglouseriv</code>	231
<code>short-long (acrstyle)</code>	202, 330	<code>\showglouserv</code>	231
<code>short-long-desc (acrstyle)</code>	204, 332	<code>\showglouservi</code>	231
<code>shortaccess (key)</code>	310	<code>sm-short-long (acrstyle)</code>	203
<code>shortplural (key)</code>	61	<code>sm-short-long-desc (acrstyle)</code>	205, 332
<code>shortpluralaccess (key)</code>	310	<code>smallcaps (option)</code>	23
<code>shotcuts (option)</code>	23	<code>smaller (option)</code>	23
<code>\showacronymlists</code>	233	<code>\SmallNewAcronymDef</code>	223, 342
<code>\showglocounter</code>	230	<code>sort (key)</code>	58
<code>\showglodesc</code>	232	<code>sort (option)</code>	10
<code>\showglodescaccess</code>	344	<code>style (option)</code>	7
<code>\showglodescplural</code>	232	<code>subentrycounter (option)</code>	9
<code>\showglodescpluralaccess</code>	344	<code>\subglossentry</code>	184
<code>\showglofirst</code>	230	<code>\subitem</code>	281
<code>\showglofirstaccess</code>	344	<code>sublistdotted (style)</code>	252
<code>\showglofirstpl</code>	230	<code>\subsubitem</code>	281
<code>\showglofirstpluralaccess</code>	344	<code>super (style)</code>	268
<code>\showgloflag</code>	233	<code>super3col (style)</code>	270
<code>\showgloindex</code>	233	<code>super3colborder (style)</code>	271
<code>\showglolevel</code>	229	<code>super3colheader (style)</code>	271
<code>\showgloloclist</code>	233	<code>super3colheaderborder (style)</code>	271
<code>\showglolong</code>	233	<code>super4col (style)</code>	272
<code>\showglolongaccess</code>	345	<code>super4colborder (style)</code>	273
<code>\showglolongpluralaccess</code>	345	<code>super4colheader (style)</code>	272
<code>\showgloname</code>	232	<code>super4colheaderborder (style)</code>	273
<code>\showglonameaccess</code>	343	<code>superborder (style)</code>	269
<code>\showgloparent</code>	229	<code>superheader (style)</code>	269
<code>\showgloplural</code>	230	<code>superheaderborder (style)</code>	270
<code>\showglopluralaccess</code>	344	<code>superragged (style)</code>	275
<code>\showgloshort</code>	233	<code>superragged3col (style)</code>	277
<code>\showgloshortaccess</code>	344	<code>superragged3colborder (style)</code>	278
<code>\showgloshortpluralaccess</code>	344	<code>superragged3colheader (style)</code>	278
<code>\showglosort</code>	232	<code>superraggedborder (style)</code>	276
<code>\showglossaries</code>	234	<code>superraggedheader (style)</code>	276
<code>\showglossarycounter</code>	234	<code>superraggedheaderborder (style)</code>	277
<code>\showglossaryentries</code>	234	<code>superraggedright3colheaderborder (style)</code>	278
<code>\showglossaryin</code>	234	<code>supertabular (environment)</code>	8, 229, 268–280
<code>\showglossaryout</code>	234	<code>supertabular package</code>	8, 229, 268, 275
<code>\showglossarytitle</code>	234	<code>symbol (key)</code>	59
<code>\showglosymbol</code>	232	<code>symbolaccess (key)</code>	310
<code>\showglosymbolaccess</code>	344	<code>\symbolname</code>	30
<code>\showglosymbolplural</code>	232	<code>symbolplural (key)</code>	59
<code>\showglosymbolpluralaccess</code>	344	<code>symbolpluralaccess (key)</code>	310
<code>\showglotext</code>	230	<code>symbols (option)</code>	25
<code>\showglotextaccess</code>	344		
<code>\showglotype</code>	230		
<code>\showglouseri</code>	231		
<code>\showglouserii</code>	231		

T	
text (key)	58
textaccess (key)	309
textcase package	3
\theequation	162
theglossary (environment) ...	5,
	16, 35, 184, 190, 266, 267, 283–286
\theHequation	162
theindex (environment)	281
toc (option)	5
\translate	30
translate (option)	21
translator package	12–14,
	26, 28, 30, 31, 165, 345, 354–357
tree (style)	282
treegroup (style)	283
treehypergroup (style)	283
treenoname (style)	284
treenonamegroup (style)	285
treenonamehypergroup (style) ..	285
type (key)	59
U	
ucmark (option)	9
W	
\warn@nomakeglossaries	151
\warn@noprintglossary	165
\writeist	33, 40, 41, 44, 143, 289, 291
X	
\xcapitalisewords	244
\xglsacccsupp	314
xindy	358
xindy	10, 23, 24, 33, 39, 42,
	44, 46, 47, 77, 103, 142, 143,
	145, 158, 161, 167, 187, 235, 290
xindy (option)	24
xindygloss (option)	24
xindynoglsnumbers (option)	24
\xmakefirstuc	243
xspace package	4, 193
user1 (key)	
user2 (key)	
user3 (key)	
user4 (key)	
user5 (key)	
user6 (key)	