

# Documented Code For glossaries v4.09

Nicola L.C. Talbot

Dickimaw Books

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

2014-08-12

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 v4.09:  $\TeX$ 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

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

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

## 1 Main Package Code

### 1.1 Package Definition

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

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries}[2014/08/12 v4.09 (NLCT)]
```

Required packages:

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

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

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

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

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

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

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

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

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

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

Check if doc has been loaded.

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

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

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

```
14 {%
```

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

```
16 }%
```

```
17 {%
```

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

```
19 }
```

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

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

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

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

```
22   \@bsphack
```

```
23   \begingroup
```

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

```
25 }
```

```
\glsorg@wrglossary
```

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

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

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

```
29   \endgroup
```

```
30   \@esphack
```

```
31 }
```

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

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

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

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

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

```
37 }
```

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

```

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

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

`\glsorg@theglossary`

```

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

`\glsorg@endtheglossary`

```

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

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

```

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

End of doc stuff.

```

55 \fi
```

## 1.2 Package Options

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

```

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

```

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

```

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

```

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

```

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

```

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

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

Determine whether or not to use numbered sections.

`\@@glossarysecstar`

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

`\@@glossaryseclabel`

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

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

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

`numberedsection`

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

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

`\@glossary@default@style`

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

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

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

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

`\@gls@declareoption`

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

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

`\glossaryentrynumbers`

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

**nonumberlist** Note that the entire number list for a given entry will be passed to `\glossaryentrynumbers` so any font changes will also be applied to the delimiters. The `nonumberlist` package option suppresses the number lists (this simply redefines `\glossaryentrynumbers` to 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@glssisacronymlist` Convenient boolean.

```
297 \newif\if@glssisacronymlist
```

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

```
298 \newcommand*\@gls@checkisacronymlist[1]{%
299   \glsIfListOfAcronyms{#1}%
300   {\@glssisacronymlisttrue}{\@glssisacronymlistfalse}%
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*\@glsacronymlists{#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 \glsetexpandfield{sortvalue}%
480 }

481 \define@choicekey[gl]{sanitize}{sort}{true,false}[true]{%
482 \setbool{glssanitizesort}{#1}%
483 \ifglssanitizesort
484 \glsetnoexpandfield{sortvalue}%
485 \else
486 \glsetexpandfield{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[gl]{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"\string~n%
1054       \expandafter\string\csname glsX#2X#1\endcsname
1055       \string" ^^J

```



```

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

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

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

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

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

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

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

```

redefinedattributes Add known attributes for all defined counters

```

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

```

```

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

```

`\@xdyuseralphabets` List of additional alphabets

```

1109 \def\@xdyuseralphabets{}

```

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

```

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

```

This code is only required for xindy:

```

1119 \ifglsxindy

```

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

```

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

```

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

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

@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   \ifglstryexists{#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   \ifglstryexists{#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   \ifglstryexists{#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\forglstryentries\expandafter
1348     [\csname glo@\@gls@thislabel @type\endcsname]
1349     {\glo@label}%
1350     {%
1351       \letcs\glo@parent{glo@\glo@label @parent}%
1352       \ifdefequal\@gls@thislabel\glo@parent
1353       {%
1354         \def\do@glshaschildren{#2}%
1355         \@endfortrue
```

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

```

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

```

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

```

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

```

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

```

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

```

First check supplied field label is defined.

```

1412   \ifdef\@glo@thisvalue
1413   {%

```

Is defined, so now check if empty.

```

1414     \ifdefempty\@glo@thisvalue
1415     {%

```

Is empty, so doesn't have field set.

```

1416       #4%
1417     }%
1418   {%

```

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

```

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

```

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

```

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

```

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

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

Is defined, so now check if empty.

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

Is empty so field hasn't been set.

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

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

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

Not defined.

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

## 1.6 Defining new glossaries

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

`\@glo@types`

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

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

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

```

Only need to do this once.

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

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

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

`\gls@doentryfmt`

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

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

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

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

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

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

`\newglossary`

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

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

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

`\ns@newglossary` Define the unstarred version.

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

Check if default has been set

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

Add this to the list of glossary types:

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

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

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

Store the file extensions:

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

Store the title:

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

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

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

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

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

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

Define sort counter if required:

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

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

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

`\altnewglossary`

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

Only define new glossaries in the preamble:

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

Only define new glossaries before `\makeglossaries`

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

`\newglossary` is used to specify the file extensions for the `makeindex` input, output and transcript files. It is written to the auxiliary file by `\newglossary`. Since it is not used by  $\text{\LaTeX}$ , `\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*\@missingnumberlist}[1]{%
1662   ??%
1663   \ifglssavenumberlist
1664     \GlossariesWarning{Missing number list for entry '#1'.
1665       Maybe makeglossaries + rerun required.}%
1666   \else
1667     \PackageError{glossaries}%
1668     {Package option 'savenumberlist=true' required.}%
1669     {%
1670       You must use the 'savenumberlist' package option
1671       to reference location lists.%
1672     }%
1673   \fi
1674 }
```

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

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

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

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

`\@gls@noexpand@field`

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

`\@gls@noexpand@fields`

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

```

```

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

```

\@gls@expand@field

```

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

```

@gls@expand@fields

```

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

```

```

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

```

startswithexpandonce

```

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

```

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

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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



```

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

```

**\newglossaryentry** For use in document environment.

```

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

```

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

```

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

```

Write key value information:

```

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

```

```

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

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

```

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

```

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

```

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

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

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

Ensure user field name is fully expanded

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

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

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

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

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

```
1843 {%
```

Found it.

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

Break out of loop.

```
1845 \@endfortrue
```

```
1846 }%
```

```
1847 {}%
```

```
1848 }%
```

```
1849 }
```

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

Allow user to add their own custom keys.

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

Starred version switches on expansion for this key.

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

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

```
1853 {%
```

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

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

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

```
1857 }%
```

```
1858 }%
```

```
1859 {}%
```

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

```
1861 }
```

Unstarred version doesn't override default expansion.

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

Check the specified key doesn't already exist.

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

```
1864 {%
```

Set up the key.

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

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

Set the default value.

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

Assignment code.

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

Define the no-link commands.

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

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

```
1874 \ifcsdef{@gls@user@#1@}%
1875 {%
1876 \PackageError{glossaries}%
1877 {Can't define '\string#5' as helper command
1878 '\expandafter\string\csname @gls@user@#1@\endcsname' already exists}%
1879 }%
1880 }%
1881 {%
1882 \expandafter\newcommand\expandafter*\expandafter
1883 {\csname @gls@user@#1\endcsname}[2][]{%
1884 \new@ifnextchar[%
1885 {\csuse{@gls@user@#1@}{##1}{##2}}%
1886 {\csuse{@gls@user@#1@}{##1}{##2}[]}}%
1887 \csdef{@gls@user@#1@}##1##2[##3]{%
1888 \@gls@field@link{##1}{##2}{#3{##2}##3}%
1889 }%
1890 \newrobustcmd*{#5}{%
1891 \expandafter\@gls@hyp@opt\csname @gls@user@#1\endcsname}%
1892 }%
```

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

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

```

1907      \@gls@field@link{##1}{##2}{#4{##2}##3}%
1908      }%
1909      \newrobustcmd*{#6}{%
1910      \expandafter\@gls@hyp@opt\csname @Gls@user@#1\endcsname}%
1911      }%

```

Finally the all caps version:

```

1912      \ifcsdef{@GLS@user@#1@}%
1913      {%
1914          \PackageError{glossaries}%
1915          {Can't define '\string#7' as helper command
1916          '\expandafter\string\csname @GLS@user@#1@\endcsname' already exists}%
1917          {}%
1918      }%
1919      {%
1920          \expandafter\newcommand\expandafter*\expandafter
1921          {\csname @GLS@user@#1\endcsname}[2][ ]{%
1922              \new@ifnextchar[%
1923                  {\csuse{@GLS@user@#1@}{##1}{##2}}%
1924                  {\csuse{@GLS@user@#1@}{##1}{##2}[ ]}}%
1925          \csdef{@GLS@user@#1@}##1##2[##3]{%
1926              \@gls@field@link{##1}{##2}{\mfirstucMakeUppercase{#3{##2}##3}}%
1927          }%
1928          \newrobustcmd*{#7}{%
1929          \expandafter\@gls@hyp@opt\csname @GLS@user@#1\endcsname}%
1930          }%
1931      }%
1932      {%
1933          \PackageError{glossaries}{Key '#1' already exists}{}%
1934      }%
1935  }

```

\gls@writedefhook

```

1936 \newcommand*{\gls@writedefhook}{}

```

\gls@assign@desc

```

1937 \newcommand*{\gls@assign@desc}[1]{%
1938   \gls@assign@field{#1}{desc}{\@glo@desc}%
1939   \gls@assign@field{\@glo@desc}{#1}{descplural}{\@glo@descplural}%
1940 }

```

\longnewglossaryentry

```

1941 \newcommand{\longnewglossaryentry}[3]{%
1942   \glsdoifnoexists{#1}%
1943   {%
1944     \bgroup
1945     \let\@org@newglossaryentryprehook\@newglossaryentryprehook
1946     \long\def\@newglossaryentryprehook{%

```

```

1947      \long\def\@glo@desc{#3\leavevmode\unskip\nopostdesc}%
1948      \@org@newglossaryentryprehook
1949  }%
1950      \renewcommand*\@gls@assign@desc}[1]{%
1951          \global\cslet{glo@glstdetoklabel{#1}@desc}{\@glo@desc}%
1952          \global\cslet{glo@glstdetoklabel{#1}@descplural}{\@glo@desc}%
1953      }
1954      \gls@defglossaryentry{#1}{#2}%
1955  \egroup
1956 }
1957 }

```

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

```
1958 \@onlypreamble{\longnewglossaryentry}
```

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

```

1959 \newcommand{\longprovideglossaryentry}[3]{%
1960   \ifglstryexists{#1}{}%
1961   {\longnewglossaryentry{#1}{#2}{#3}}%
1962 }
1963 \@onlypreamble{\longprovideglossaryentry}

```

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

Defines a new entry without checking if it already exists.

```
1964 \newcommand{\gls@defglossaryentry}[2]{%
```

Store label

```
1965   \edef\@glo@label{\glstdetoklabel{#1}}%
```

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

```
1966   \let\glslabel\@glo@label
```

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

```
1967   \let\@glo@name\@glsnname
```

```
1968   \let\@glo@desc\@glsnodesc
```

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

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

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

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

```
1973   \let\@glo@text\@gls@default@value
```

```
1974   \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.)

```
1975 \let\@glo@first\@gls@default@value
1976 \let\@glo@firstplural\@gls@default@value
```

Set the default sort:

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

Set the default counter:

```
1978 \let\@glo@counter\@gls@default@value
```

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

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

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

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

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

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

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

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

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

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

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

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

```
1991 \def\@glo@longpl{}%
```

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

```
1992 \@newglossaryentryprehook
```

Extract key-val information from third parameter:

```
1993 \setkeys{glossentry}{#2}%
```

Check there is a default glossary.

```
1994 \ifundef\glsdefaulttype
```

```
1995 {%
```

```
1996 \PackageError{glossaries}%
```

```
1997 {No default glossary type (have you used ‘nomain’?)}%
```

```
1998 {If you use package option ‘nomain’ you must define
```

```
1999 a new glossary before you can define entries}%
```

```
2000 }%
```

```
2001 {}%
```

Assign type. This must be fully expandable

```
2002 \gls@assign@field{\glsdefaulttype}{\@glo@label}{type}{\@glo@type}%
```

```
2003 \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.

```

2004 \ifcsundef{glolist@\@glo@type}%
2005 {%
2006     \PackageError{glossaries}%
2007     {Glossary type ‘\@glo@type’ has not been defined}%
2008     {You need to define a new glossary type, before making entries
2009     in it}%
2010 }%
2011 {%

```

Check if it's an ignored glossary

```

2012 \ifignoredglossary\@glo@type
2013 {%

```

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

```

2014 \ifx\@glo@desc\glsnodel
2015 \let\@glo@desc\empty
2016 \fi
2017 }%
2018 {%
2019 }%
2020 \protected@edef\@glolist@{\csname glolist@\@glo@type\endcsname}%
2021 \expandafter\xdef\csname glolist@\@glo@type\endcsname{%
2022 \@glolist@{\@glo@label},}%
2023 }%

```

Initialise level to 0.

```

2024 \gls@level=0\relax

```

Has this entry been assigned a parent?

```

2025 \ifx\@glo@parent\empty

```

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

```

2026 \expandafter\gdef\csname glo@\@glo@label @parent\endcsname{}%
2027 \else

```

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

```

2028 \ifdefequal\@glo@label\@glo@parent%
2029 {%
2030     \PackageError{glossaries}{Entry ‘\@glo@label’ can't be its own parent}{}%
2031     \def\@glo@parent{}%
2032     \expandafter\gdef\csname glo@\@glo@label @parent\endcsname{}%
2033 }%
2034 {%

```

Check the parent exists:

```

2035 \ifglentryexists{\@glo@parent}%
2036 {%

```

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

```

2037 \expandafter\xdef\csname glo@\@glo@label @parent\endcsname{%
2038 \@glo@parent}%

```



Determine level.

```
2039      \gls@level=\csname glo@\@glo@parent @level\endcsname\relax
2040      \advance\gls@level by 1\relax
```

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

```
2041      \ifx\@glo@name\@gls@name
2042      \expandafter\let\expandafter\@glo@name
2043      \csname glo@\@glo@parent @name\endcsname
```

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

```
2044      \ifx\@glo@plural\@gls@default@value
2045      \expandafter\let\expandafter\@glo@plural
2046      \csname glo@\@glo@parent @plural\endcsname
2047      \fi
2048      \fi
2049      }%
2050      {%
```

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

```
2051      \PackageError{glossaries}%
2052      {%
2053      Invalid parent '\@glo@parent'
2054      for entry '\@glo@label' - parent doesn't exist%
2055      }%
2056      {%
2057      Parent entries must be defined before their children%
2058      }%
2059      \def\@glo@parent{%
2060      \expandafter\gdef\csname glo@\@glo@label @parent\endcsname{%
2061      }%
2062      }%
2063      \fi
```

Set the level for this entry

```
2064      \expandafter\xdef\csname glo@\@glo@label @level\endcsname{\number\gls@level}%
```

Define commands associated with this entry:

```
2065      \gls@assign@field{\@glo@name}{\@glo@label}{sortvalue}{\@glo@sort}%
2066      \letcs\@glo@sort{glo@\@glo@label @sortvalue}%
2067      \gls@assign@field{\@glo@name}{\@glo@label}{text}{\@glo@text}%
2068      \expandafter\gls@assign@field\expandafter
2069      {\csname glo@\@glo@label @text\endcsname\glspluralsuffix}%
2070      {\@glo@label}{plural}{\@glo@plural}%
2071      \expandafter\gls@assign@field\expandafter
2072      {\csname glo@\@glo@label @text\endcsname}%
2073      {\@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.

```
2074      \ifx\@glo@first\@gls@default@value
```

```

2075 \expandafter\gls@assign@field\expandafter
2076 {\csname glo@\@glo@label @plural\endcsname}%
2077 {\@glo@label}{firstpl}{\@glo@firstplural}%
2078 \else
2079 \expandafter\gls@assign@field\expandafter
2080 {\csname glo@\@glo@label @first\endcsname\glspluralsuffix}%
2081 {\@glo@label}{firstpl}{\@glo@firstplural}%
2082 \fi

2083 \ifcsundef{@glo@type@\@glo@type @counter}%
2084 {%
2085 \def\@glo@defaultcounter{\glscounter}%
2086 }%
2087 {%
2088 \letcs\@glo@defaultcounter{@glo@type@\@glo@type @counter}%
2089 }%
2090 \gls@assign@field{\@glo@defaultcounter}{\@glo@label}{counter}{\@glo@counter}%
2091 \gls@assign@field{}{\@glo@label}{useri}{\@glo@useri}%
2092 \gls@assign@field{}{\@glo@label}{userii}{\@glo@userii}%
2093 \gls@assign@field{}{\@glo@label}{useriii}{\@glo@useriii}%
2094 \gls@assign@field{}{\@glo@label}{useriv}{\@glo@useriv}%
2095 \gls@assign@field{}{\@glo@label}{userv}{\@glo@userv}%
2096 \gls@assign@field{}{\@glo@label}{uservi}{\@glo@uservi}%
2097 \gls@assign@field{}{\@glo@label}{short}{\@glo@short}%
2098 \gls@assign@field{}{\@glo@label}{shortpl}{\@glo@shortpl}%
2099 \gls@assign@field{}{\@glo@label}{long}{\@glo@long}%
2100 \gls@assign@field{}{\@glo@label}{longpl}{\@glo@longpl}%
2101 \ifx\@glo@name\@glsnname
2102 \@glsnname
2103 \let\@glo@name\@gls@default@value
2104 \fi
2105 \gls@assign@field{}{\@glo@label}{name}{\@glo@name}%

```

Set default numberlist if not defined:

```

2106 \ifcsundef{glo@\@glo@label @numberlist}%
2107 {%
2108 \csxdef{glo@\@glo@label @numberlist}{%
2109 \noexpand\@gls@missingnumberlist{\@glo@label}}%
2110 }%
2111 {}%

```

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.

```

2112 \def\@glo@@desc{\@glo@first}%
2113 \ifx\@glo@desc\@glo@@desc
2114 \let\@glo@desc\@glo@first
2115 \fi
2116 \ifx\@glo@desc\@glsnname
2117 \@glsnname

```

```

2118     \let\@glodesc\@gls@default@value
2119     \fi
2120     \gls@assign@desc{\@glo@label}%

```

Set the sort key for this entry:

```

2121     \@gls@defsort{\@glo@type}{\@glo@label}%

2122     \def\@glo@@symbol{\@glo@text}%
2123     \ifx\@glo@symbol\@glo@@symbol
2124         \let\@glo@symbol\@glo@text
2125     \fi
2126     \gls@assign@field{\relax}{\@glo@label}{symbol}{\@glo@symbol}%
2127     \expandafter
2128         \gls@assign@field\expandafter
2129         {\csname glo@\@glo@label @symbol\endcsname}
2130         {\@glo@label}{symbolplural}{\@glo@symbolplural}%

```

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

```

2131     \expandafter\xdef\csname glo@\@glo@label @flagfalse\endcsname{%
2132         \noexpand\global
2133         \noexpand\let\expandafter\noexpand
2134             \csname ifglo@\@glo@label @flag\endcsname\noexpand\iffalse
2135     }%
2136     \expandafter\xdef\csname glo@\@glo@label @flagtrue\endcsname{%
2137         \noexpand\global
2138         \noexpand\let\expandafter\noexpand
2139             \csname ifglo@\@glo@label @flag\endcsname\noexpand\iftrue
2140     }%
2141     \csname glo@\@glo@label @flagfalse\endcsname

```

Sort out any cross-referencing if required.

```

2142     \ifdefvoid\@glo@see
2143     {}%
2144     {%
2145         \protected@edef\@do@glsee{%
2146             \noexpand\@gls@fixbraces\noexpand\@glo@list\@glo@see
2147             \noexpand\@nil
2148             \noexpand\expandafter\noexpand\@glsee\noexpand\@glo@list{\@glo@label}}%
2149         \@do@glsee
2150     }%

```

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

```

2151     \ifignoredglossary\@glo@type
2152     {%
2153         \csdef{glo@\@glo@label @index}{}%
2154     }
2155     {%
2156         \do@glo@storeentry{\@glo@label}%
2157     }%

```

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

```
2158 \@newglossaryentryposthook
2159 }
```

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

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

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

```
2161 \newcommand*{\@newglossaryentryposthook}{}%
```

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

```
2162 \newcommand*{\glsmoveentry}[2]{%
2163   \edef\@glo@thislabel{\glsdetoklabel{#1}}%
2164   \edef\glo@type{\csname glo@\@glo@thislabel @type\endcsname}%
2165   \def\glo@list{,%}%
2166   \for\glseentries[\glo@type]{\glo@label}%
2167   {%
2168     \ifdefequal\@glo@thislabel\glo@label
2169     {}{\eappto\glo@list{\glo@label,%}}%
2170   }%
2171   \cslet\glolist@\glo@type{\glo@list}%
2172   \csdef\glo@\@glo@thislabel @type{#2}%
2173 }
```

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

```
2174 \ifglxindy
2175   \newcommand*{\@glossaryentryfield}{\string\glossentry}
2176 \else
2177   \newcommand*{\@glossaryentryfield}{\string\glossentry}
2178 \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.)

```
2179 \ifglxindy
2180   \newcommand*{\@glossarysubentryfield}{%
2181     \string\subglossentry}
2182 \else
2183   \newcommand*{\@glossarysubentryfield}{%
2184     \string\subglossentry}
2185 \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.)

```
2186 \newcommand{\@glo@storeentry}[1]{%
```

Escape makeindex/xindy special characters in the label:

```
2187 \edef\@glo@esclabel{#1}%
2188 \@gls@checkmkidxchars\@glo@esclabel
```

Get the sort string and escape any special characters

```
2189 \protected@edef\@glo@sort{\csname glo@#1@sort\endcsname}%
2190 \@gls@checkmkidxchars\@glo@sort
```

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

```
2191 \@gls@checkmkidxchars\@glo@prefix
```

Get the parent, if one exists

```
2192 \edef\@glo@parent{\csname glo@#1@parent\endcsname}%
```

Write the information to the glossary file.

```
2193 \ifglsxindy
```

Store using xindy syntax.

```
2194 \ifx\@glo@parent\@empty
```

Entry doesn't have a parent

```
2195 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
2196 (\string"\@glo@sort\string" %
2197 \string"\@glo@prefix\@glossaryentryfield{\@glo@esclabel}\string") %
2198 }%
2199 \else
```

Entry has a parent

```
2200 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
2201 \csname glo@\@glo@parent @index\endcsname
2202 (\string"\@glo@sort\string" %
2203 \string"\@glo@prefix\@glossarysubentryfield
2204 {\csname glo@#1@level\endcsname}{\@glo@esclabel}\string") %
2205 }%
2206 \fi
2207 \else
```

Store using makeindex syntax.

```
2208 \ifx\@glo@parent\@empty
```

Sanitize \@glo@prefix

```
2209 \@onelevel@sanitize\@glo@prefix
```

Entry doesn't have a parent

```
2210 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
2211 \@glo@sort\@gls@actualchar\@glo@prefix
2212 \@glossaryentryfield{\@glo@esclabel}%
```

```

2213     }%
2214     \else
      Entry has a parent
2215     \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
2216       \csname glo@\@glo@parent @index\endcsname\@gls@levelchar
2217       \@glo@sort\@gls@actualchar\@glo@prefix
2218       \@glossarysubentryfield
2219       {\csname glo@#1@level\endcsname}\@glo@esclabel}%
2220     }%
2221     \fi
2222 \fi
2223 }

```

## 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`

```

2224 \AtBeginDocument{%
2225   \ifpackageloaded{amsmath}%
2226   {\let\gls@ifnotmeasuring\@gls@ifnotmeasuring}%
2227   {}}%
2228 }
2229 \newcommand*{\@gls@ifnotmeasuring}[1]{%
2230   \ifmeasuring@
2231   \else
2232     #1%
2233   \fi
2234 }
2235 \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.

```

2236 \newcommand*{\glsreset}[1]{%
2237   \gls@ifnotmeasuring
2238   {%
2239     \glsdoifexists{#1}%
2240     {%
2241       \expandafter\global\csname glo@\glsdetoklabel{#1}@flagfalse\endcsname
2242     }%
2243   }%
2244 }

```

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

```
2245 \newcommand*{\glslocalreset}[1]{%
2246   \gls@ifnotmeasuring
2247   {%
2248     \glsdoifexists{#1}%
2249     {%
2250       \expandafter\let\csname ifglo@\glsdetoklabel{#1}@flag\endcsname\iffalse
2251     }%
2252   }%
2253 }
```

`\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.

```
2254 \newcommand*{\glsunset}[1]{%
2255   \gls@ifnotmeasuring
2256   {%
2257     \glsdoifexists{#1}%
2258     {%
2259       \expandafter\global\csname glo@\glsdetoklabel{#1}@flagtrue\endcsname
2260     }%
2261   }%
2262 }
```

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

```
2263 \newcommand*{\glslocalunset}[1]{%
2264   \gls@ifnotmeasuring
2265   {%
2266     \glsdoifexists{#1}%
2267     {%
2268       \expandafter\let\csname ifglo@\glsdetoklabel{#1}@flag\endcsname\iftrue
2269     }%
2270   }%
2271 }
```

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

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

`\glsresetall`

```
2272 \newcommand*{\glsresetall}[1][\@glo@types]{%
2273   \forallglsentries[#1]{\@glsentry}%
2274   {%
2275     \glsreset{\@glsentry}%
2276   }%
2277 }
```

As above, but with only a local effect:

`\glslocalresetall`

```
2278 \newcommand*{\glslocalresetall}[1][\@glo@types]{%
```

```

2279 \forallglsentries[#1]{\@glsentry}%
2280 {%
2281     \glslocalreset{\@glsentry}%
2282 }%
2283 }

```

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

`\glsunsetall`

```

2284 \newcommand*{\glsunsetall}[1][\@gls@types]{%
2285     \forallglsentries[#1]{\@glsentry}%
2286     {%
2287         \glsunset{\@glsentry}%
2288     }%
2289 }

```

As above, but with only a local effect:

`\glslocalunsetall`

```

2290 \newcommand*{\glslocalunsetall}[1][\@gls@types]{%
2291     \forallglsentries[#1]{\@glsentry}%
2292     {%
2293         \glslocalunset{\@glsentry}%
2294     }%
2295 }

```

## 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.<sup>1</sup>

`\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`

```

2296 \newcommand*{\loadglsentries}[2][\@gls@default]{%
2297     \let\@gls@default\glsdefaulttype
2298     \def\glsdefaulttype{#1}\input{#2}%
2299     \let\glsdefaulttype\@gls@default
2300 }

```

---

<sup>1</sup>and any other valid  $\LaTeX$  code that can be used in the preamble.



`\loadglsentries` can only be used in the preamble:  
2301 `\@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`

2302 `\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

2303 `\newcommand*{\glsentryfmt}{%`

2304 `\@@gls@default@entryfmt\glsdisplayfirst\glsdisplay`

2305 `}`

Format that provides backwards compatibility:

2306 `\newcommand*{\@@gls@default@entryfmt}[2]{%`

2307 `\ifdefempty\glscustomtext`

2308 `{%`

2309 `\glsifplural`

2310 `{%`

Plural form

2311 `\glscapscase`

2312 `{%`

Don’t adjust case

2313 `\ifglsused\glslabel`

2314 `{%`

Subsequent use

2315 `#2{\glsentryplural{\glslabel}}%`

2316 `{\glsentrydescplural{\glslabel}}%`

2317 `{\glsentrysymbolplural{\glslabel}}{\glsinsert}}%`

2318 `}%`

2319 `{%`

First use

2320 `#1{\glsentryfirstplural{\glslabel}}%`

2321 `{\glsentrydescplural{\glslabel}}%`

2322 `{\glsentrysymbolplural{\glslabel}}{\glsinsert}}%`

```

2323     }%
2324     }%
2325     {%

```

Make first letter upper case

```

2326     \ifglused\glslabel
2327     {%

```

Subsequent use. (Expansion was used in version 3.07 and below in case the name wasn't the first thing to be displayed, but now the user can sort out the upper casing in `\defglentryfmt`, which avoids the issues caused by fragile commands.)

```

2328     \ifbool{glscompatible-3.07}%
2329     {%
2330     \protected@edef\@glo@etext{%
2331     #2{\glentryplural{\glslabel}}%
2332     {\glentrydescplural{\glslabel}}%
2333     {\glentrysymbolplural{\glslabel}}{\glsinsert}}%
2334     \xmakefirstuc\@glo@etext
2335     }%
2336     {%
2337     #2{\Glsentryplural{\glslabel}}%
2338     {\glentrydescplural{\glslabel}}%
2339     {\glentrysymbolplural{\glslabel}}{\glsinsert}%
2340     }%
2341     }%
2342     {%

```

First use

```

2343     \ifbool{glscompatible-3.07}%
2344     {%
2345     \protected@edef\@glo@etext{%
2346     #1{\glentryfirstplural{\glslabel}}%
2347     {\glentrydescplural{\glslabel}}%
2348     {\glentrysymbolplural{\glslabel}}{\glsinsert}}%
2349     \xmakefirstuc\@glo@etext
2350     }%
2351     {%
2352     #1{\Glsentryfirstplural{\glslabel}}%
2353     {\glentrydescplural{\glslabel}}%
2354     {\glentrysymbolplural{\glslabel}}{\glsinsert}%
2355     }%
2356     }%
2357     }%
2358     {%

```

Make all upper case

```

2359     \ifglused\glslabel
2360     {%

```

Subsequent use

```

2361      \mfirstucMakeUppercase{#2{\glsentryplural{\glslabel}}}%
2362      {\glsentrydescplural{\glslabel}}}%
2363      {\glsentrysymbolplural{\glslabel}}{\glsinsert}}}%
2364  }%
2365  {%

```

#### First use

```

2366      \mfirstucMakeUppercase{#1{\glsentryfirstplural{\glslabel}}}%
2367      {\glsentrydescplural{\glslabel}}}%
2368      {\glsentrysymbolplural{\glslabel}}{\glsinsert}}}%
2369  }%
2370  }%
2371  }%
2372  {%

```

#### Singular form

```

2373      \glscapscase
2374      {%

```

#### Don't adjust case

```

2375      \ifglused\glslabel
2376      {%

```

#### Subsequent use

```

2377      #2{\glsentrytext{\glslabel}}}%
2378      {\glsentrydesc{\glslabel}}}%
2379      {\glsentrysymbol{\glslabel}}{\glsinsert}%
2380  }%
2381  {%

```

#### First use

```

2382      #1{\glsentryfirst{\glslabel}}}%
2383      {\glsentrydesc{\glslabel}}}%
2384      {\glsentrysymbol{\glslabel}}{\glsinsert}%
2385  }%
2386  }%
2387  {%

```

#### Make first letter upper case

```

2388      \ifglused\glslabel
2389      {%

```

#### Subsequent use

```

2390      \ifbool{glscompatible-3.07}%
2391      {%
2392      \protected@edef\@glo@etext{%
2393      #2{\glsentrytext{\glslabel}}}%
2394      {\glsentrydesc{\glslabel}}}%
2395      {\glsentrysymbol{\glslabel}}{\glsinsert}}}%
2396      \xmakefirstuc\@glo@etext
2397  }%
2398  {%

```

```

2399         #2{\Glsentrytext{\glslabel}}%
2400         {\glsentrydesc{\glslabel}}%
2401         {\glsentrysymbol{\glslabel}}{\glsinsert}%
2402     }%
2403 }%
2404 {%

    First use
2405     \ifbool{glscompatible-3.07}%
2406     {%
2407         \protected@edef\@glo@etext{%
2408             #1{\glsentryfirst{\glslabel}}%
2409             {\glsentrydesc{\glslabel}}%
2410             {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2411         \xmakefirstuc\@glo@etext
2412     }%
2413     {%
2414         #1{\Glsentryfirst{\glslabel}}%
2415         {\glsentrydesc{\glslabel}}%
2416         {\glsentrysymbol{\glslabel}}{\glsinsert}%
2417     }%
2418 }%
2419 }%
2420 {%

    Make all upper case
2421     \ifglsused\glslabel
2422     {%

        Subsequent use
2423         \mfirstucMakeUppercase{#2{\glsentrytext{\glslabel}}%
2424         {\glsentrydesc{\glslabel}}%
2425         {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2426     }%
2427     {%

        First use
2428         \mfirstucMakeUppercase{#1{\glsentryfirst{\glslabel}}%
2429         {\glsentrydesc{\glslabel}}%
2430         {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2431     }%
2432 }%
2433 }%
2434 }%
2435 {%

    Custom text provided in \glsdisp
2436     \ifglsused{\glslabel}%
2437     {%

        Subsequent use
2438         #2{\glscustomtext}%

```

```

2439      {\glsentrydesc{\glslabel}}}%
2440      {\glsentrysymbol{\glslabel}}{}}%
2441    }%
2442    {%

```

First use

```

2443      #1{\glscustomtext}%
2444      {\glsentrydesc{\glslabel}}}%
2445      {\glsentrysymbol{\glslabel}}{}}%
2446    }%
2447    }%
2448 }

```

`\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.

```

2449 \newcommand*{\glsgenentryfmt}{%
2450   \ifdefempty\glscustomtext
2451   {%
2452     \glsifplural
2453     {%

```

Plural form

```

2454     \glscapscase
2455     {%

```

Don't adjust case

```

2456     \ifglsused\glslabel
2457     {%

```

Subsequent use

```

2458     \glsentryplural{\glslabel}\glsinsert
2459     }%
2460     {%

```

First use

```

2461     \glsentryfirstplural{\glslabel}\glsinsert
2462     }%
2463     }%
2464     {%

```

Make first letter upper case

```

2465     \ifglsused\glslabel
2466     {%

```

Subsequent use.

```

2467     \Glsentryplural{\glslabel}\glsinsert
2468     }%
2469     {%

```

First use

```

2470     \Glsentryfirstplural{\glslabel}\glsinsert
2471     }%

```

```

2472     }%
2473     {%

    Make all upper case
2474         \ifglused\glslabel
2475         {%

    Subsequent use
2476         \mfirstucMakeUppercase
2477         {\glentryplural{\glslabel}\glsinsert}%
2478     }%
2479     {%

    First use
2480         \mfirstucMakeUppercase
2481         {\glentryfirstplural{\glslabel}\glsinsert}%
2482     }%
2483     }%
2484     }%
2485     {%

    Singular form
2486     \glscapscase
2487     {%

    Don't adjust case
2488     \ifglused\glslabel
2489     {%

    Subsequent use
2490     \glentrytext{\glslabel}\glsinsert
2491     }%
2492     {%

    First use
2493     \glentryfirst{\glslabel}\glsinsert
2494     }%
2495     }%
2496     {%

    Make first letter upper case
2497     \ifglused\glslabel
2498     {%

    Subsequent use
2499     \Glentrytext{\glslabel}\glsinsert
2500     }%
2501     {%

    First use
2502     \Glentryfirst{\glslabel}\glsinsert
2503     }%
2504     }%
2505     {%

```

Make all upper case

```
2506      \ifglsused\glslabel
2507      {%
```

Subsequent use

```
2508      \mfirstucMakeUppercase{\glsentrytext{\glslabel}\glsinsert}%
2509      }%
2510      {%
```

First use

```
2511      \mfirstucMakeUppercase{\glsentryfirst{\glslabel}\glsinsert}%
2512      }%
2513      }%
2514      }%
2515      }%
2516      {%
```

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

```
2517      \glscustomtext\glsinsert
2518      }%
2519 }
```

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

```
2520 \newcommand*{\glsngenacfmt}{%
2521   \ifdefempty\glscustomtext
2522   {%
2523     \ifglsused\glslabel
2524     {%
```

Subsequent use:

```
2525     \glsifplural
2526     {%
```

Subsequent plural form:

```
2527     \glscapscase
2528     {%
```

Subsequent plural form, don't adjust case:

```
2529     \acronymfont{\glsentryshortpl{\glslabel}}\glsinsert
2530     }%
2531     {%
```

Subsequent plural form, make first letter upper case:

```
2532     \acronymfont{\Glsentryshortpl{\glslabel}}\glsinsert
2533     }%
2534     {%
```

Subsequent plural form, all caps:

```
2535     \mfirstucMakeUppercase
2536     {\acronymfont{\glsentryshortpl{\glslabel}}\glsinsert}%
```

2537       }%  
 2538       }%  
 2539       {%

Subsequent singular form

2540       \glscapscase  
 2541       {%

Subsequent singular form, don't adjust case:

2542       \acronymfont{\glstryshort{\glslabel}}\glsinsert  
 2543       }%  
 2544       {%

Subsequent singular form, make first letter upper case:

2545       \acronymfont{\Glsstryshort{\glslabel}}\glsinsert  
 2546       }%  
 2547       {%

Subsequent singular form, all caps:

2548       \mfirstucMakeUppercase  
 2549       {\acronymfont{\glstryshort{\glslabel}}\glsinsert}%  
 2550       }%  
 2551       }%  
 2552       }%  
 2553       {%

First use:

2554       \glsifplural  
 2555       {%

First use plural form:

2556       \glscapscase  
 2557       {%

First use plural form, don't adjust case:

2558       \genplacrfullformat{\glslabel}{\glsinsert}%  
 2559       }%  
 2560       {%

First use plural form, make first letter upper case:

2561       \Genplacrfullformat{\glslabel}{\glsinsert}%  
 2562       }%  
 2563       {%

First use plural form, all caps:

2564       \mfirstucMakeUppercase  
 2565       {\genplacrfullformat{\glslabel}{\glsinsert}}%  
 2566       }%  
 2567       }%  
 2568       {%

First use singular form

2569       \glscapscase  
 2570       {%



First use singular form, don't adjust case:

```
2571      \genacrfullformat{\glslabel}{\glsinsert}%
2572      }%
2573      {%
```

First use singular form, make first letter upper case:

```
2574      \Genacrfullformat{\glslabel}{\glsinsert}%
2575      }%
2576      {%
```

First use singular form, all caps:

```
2577      \mfirstucMakeUppercase
2578      {\genacrfullformat{\glslabel}{\glsinsert}}%
2579      }%
2580      }%
2581      }%
2582      }%
2583      {%
```

User supplied text.

```
2584      \glscustomtext
2585      }%
2586 }
```

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

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

```
2587 \newcommand*{\genacrfullformat}[2]{%
2588   \glsentrylong{#1}#2\space
2589   (\protect\firstacronymfont{\glsentryshort{#1}})%
2590 }
```

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

As above but makes the first letter upper case.

```
2591 \newcommand*{\Genacrfullformat}[2]{%
2592   \protected@edef\gls@text{\genacrfullformat{#1}{#2}}%
2593   \xmakefirstuc\gls@text
2594 }
```

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

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

```
2595 \newcommand*{\genplacrfullformat}[2]{%
2596   \glsentrylongpl{#1}#2\space
2597   (\protect\firstacronymfont{\glsentryshortpl{#1}})%
```

2598 }

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

As above but makes the first letter upper case.

```
2599 \newcommand*{\Genplacrfullformat}[2]{%
2600   \protected@edef\gls@text{\genplacrfullformat{#1}{#2}}%
2601   \xmakefirstuc\gls@text
2602 }
```

`\glsdisplayfirst` Deprecated. Kept for backward compatibility.

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

`\glsdisplay` Deprecated. Kept for backward compatibility.

```
2604 \newcommand*{\glsdisplay}[4]{#1#4}
```

`\defglsdisplay` Deprecated. Kept for backward compatibility.

```
2605 \newcommand*{\defglsdisplay}[2][\glsdefaulttype]{%
2606   \GlossariesWarning{\string\defglsdisplay\space is now obsolete.^^J
2607   Use \string\defglsentryfmt\space instead}%
2608   \expandafter\def\csname gls@#1@display\endcsname##1##2##3##4{#2}%
2609   \edef\@gls@doentrydef{%
2610     \noexpand\defglsentryfmt[#1]{%
2611       \noexpand\ifcsdef{gls@#1@displayfirst}%
2612       {%
2613         \noexpand\@gls@default@entryfmt
2614         {\noexpand\csuse{gls@#1@displayfirst}}%
2615         {\noexpand\csuse{gls@#1@display}}%
2616       }%
2617       {%
2618         \noexpand\@gls@default@entryfmt
2619         {\noexpand\glsdisplayfirst}%
2620         {\noexpand\csuse{gls@#1@display}}%
2621       }%
2622     }%
2623   }%
2624   \@gls@doentrydef
2625 }
```

`\defglsdisplayfirst` Deprecated. Kept for backward compatibility.

```
2626 \newcommand*{\defglsdisplayfirst}[2][\glsdefaulttype]{%
2627   \GlossariesWarning{\string\defglsdisplayfirst\space is now obsolete.^^J
2628   Use \string\defglsentryfmt\space instead}%
2629   \expandafter\def\csname gls@#1@displayfirst\endcsname##1##2##3##4{#2}%
2630   \edef\@gls@doentrydef{%
2631     \noexpand\defglsentryfmt[#1]{%
2632       \noexpand\ifcsdef{gls@#1@display}%

```

```

2633      {%
2634      \noexpand\@gls@default@entryfmt
2635      {\noexpand\csuse{gls@#1@displayfirst}}}%
2636      {\noexpand\csuse{gls@#1@display}}}%
2637      }%
2638      {%

2639      \noexpand\@gls@default@entryfmt
2640      {\noexpand\csuse{gls@#1@displayfirst}}}%
2641      {\noexpand\glsdisplay}}}%
2642      }%
2643      }%
2644      }%
2645      \@gls@doentrydef
2646      }

```

### 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.

```

2647 \define@key{glslink}{counter}{%
2648   \ifcsundef{c@#1}%
2649   {%
2650     \PackageError{glossaries}%
2651     {There is no counter called ‘#1’}%
2652     {%
2653       The counter key should have the name of a valid counter
2654       as its value%
2655     }%
2656   }%
2657   {%
2658     \def\@gls@counter{#1}%
2659   }%
2660 }

```

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.

```
2661 \define@key{glslink}{format}{%
2662   \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.

```
2663 \define@boolkey{glslink}{hyper}[true]{}
```

Initialise hyper key.

```
2664 \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.

```
2665 \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.

```
2666 \newcommand*{\glslinkvar}[3]{#1}
```

\glsifhyper    Now deprecated.

```
2667 \newcommand*{\glsifhyper}[2]{%
2668   \glslinkvar{#1}{#2}{#1}%
2669   \GlossariesWarning{\string\glsifhyper\space is deprecated. Did
2670     you mean \string\glsifhyperon\space or \string\glslinkvar?}%
2671 }
```

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

```
2672 \newcommand*{\@gls@hyp@opt}[1]{%
2673   \let\glslinkvar\@firstofthree
2674   \let\@gls@hyp@opt@cs#1\relax
2675   \ifstar{\s@gls@hyp@opt}%
2676   {\@ifnextchar+{\@firstoftwo{\p@gls@hyp@opt}}{#1}}%
2677 }
```

\s@gls@hyp@opt    Starred version

```
2678 \newcommand*{\s@gls@hyp@opt}[1][ ]{%
```

```

2679 \let\glslinkvar\@secondofthree
2680 \@gls@hyp@opt@cs[hyper=false,#1]}

```

\p@gls@hyp@opt Plus version

```

2681 \newcommand*{\p@gls@hyp@opt}[1] [] {%
2682 \let\glslinkvar\@thirdofthree
2683 \@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

```

2684 \newrobustcmd*{\glslink}{%
2685 \@gls@hyp@opt\@gls@@link
2686 }

```

\@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.

```

2687 \newcommand*{\@gls@@link}[3] [] {%
2688 \ifglsentryexists{#2}%
2689 {%
2690 \let\do@gls@link@checkfirsthyper\relax
2691 \@gls@link[#1]{#2}{#3}%
2692 }{%
2693 \PackageError{glossaries}{Glossary entry ‘#2’ has not been
2694 defined}{You need to define a glossary entry before you
2695 can use it.}%

```

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

```

2696 \glstextformat{#3}%
2697 }%
2698 }

```

\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`.

```

2699 \newcommand*{\@gls@link@checkfirsthyper}{%
2700   \ifglsused{\glslabel}%
2701   {%
2702   }%
2703   {%
2704     \gls@checkisacronymlist\glstype
2705     \ifglshyperfirst
2706       \ifglsisacronymlist
2707         \ifglsacrfootnote
2708           \KV@glslink@hyperfalse
2709         \fi
2710       \fi
2711     \else
2712       \KV@glslink@hyperfalse
2713     \fi
2714   }%

```

Allow user to hook into this

```

2715 \glslinkcheckfirsthyperhook
2716 }

```

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

```

2717 \newcommand*{\glslinkcheckfirsthyperhook}{}

```

`\@gls@link`

```

2718 \def\@gls@link[#1]#2#3{%

```

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

```

2719   \leavevmode
2720   \edef\glslabel{\glsdetoklabel{#2}}%

```

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

```

2721   \def\@gls@link@opts{#1}%
2722   \let\@gls@link@label\glslabel

```

```

2723   \def\@glsnumberformat{\glsnumberformat}%
2724   \edef\@gls@counter{\csname glo@\glslabel @counter\endcsname}%

```

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

```

2725   \edef\glstype{\csname glo@\glslabel @type\endcsname}%

```

Save original setting

```

2726   \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper

```

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

```

2727   \expandafter\DTLifinlist\expandafter
2728   {\glstype}{\@gls@nohyperlist}%

```

```

2729    {%
2730        \KV@glslink@hyperfalse
2731    }%
2732    {%
2733    }%

```

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`.

```

2734    \do@glslink@checkfirsthyper
2735    \setkeys{glslink}{#1}%

    Define \glsifhyperon
2736    \ifKV@glslink@hyper
2737        \let\glsifhyperon\@firstoftwo
2738    \else
2739        \let\glsifhyperon\@secondoftwo
2740    \fi

```

Store the entry's counter in `\theglentrycounter`

```

2741    \@gls@saveentrycounter

```

Define sort key if necessary:

```

2742    \@gls@setsort{\glslabel}%

    (De-tok'ing done by \@do@wrglossary)
2743    \@do@wrglossary{#2}%
2744    \ifKV@glslink@hyper
2745        \@glslink{\glolinkprefix\glslabel}{\glstextformat{#3}}%
2746    \else

2747        \glstextformat{#3}%
2748    \fi

```

Restore original setting

```

2749    \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper
2750 }

```

`\glolinkprefix`

```

2751 \newcommand*{\glolinkprefix}{glo:}

```

`\glentrycounter` Set default value of entry counter

```

2752 \def\glentrycounter{\glscounter}%

```

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

```

2753 \newcommand*{\@gls@saveentrycounter}{%
2754     \def\@gls@Hcounter{}%

    Are we using equation counter?
2755     \ifthenelse{\equal{\@gls@counter}{equation}}{%
2756     {

```

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

```

2757 \ifcsundef{xatlevel@}%
2758 {%
2759 \edef\theglsentrycounter{\expandafter\noexpand
2760 \csname the\@gls@counter\endcsname}%
2761 }%
2762 {%
2763 \ifx\xatlevel@\@empty
2764 \edef\theglsentrycounter{\expandafter\noexpand
2765 \csname the\@gls@counter\endcsname}%
2766 \else
2767 \savecounters@
2768 \advance\c@equation by 1\relax
2769 \edef\theglsentrycounter{\csname the\@gls@counter\endcsname}%

```

Check if hyperref version of this counter

```

2770 \ifcsundef{theH\@gls@counter}%
2771 {%
2772 \def\@gls@Hcounter{\theglsentrycounter}%
2773 }%
2774 {%
2775 \def\@gls@Hcounter{\csname theH\@gls@counter\endcsname}%
2776 }%
2777 \protected@edef\theHglentrycounter{\@gls@Hcounter}%
2778 \restorecounters@
2779 \fi
2780 }%
2781 }%
2782 {%

```

Not using equation counter so no special measures:

```

2783 \edef\theglsentrycounter{\expandafter\noexpand
2784 \csname the\@gls@counter\endcsname}%
2785 }%

```

Check if hyperref version of this counter

```

2786 \ifx\@gls@Hcounter\@empty
2787 \ifcsundef{theH\@gls@counter}%
2788 {%
2789 \def\theHglentrycounter{\theglsentrycounter}%
2790 }%
2791 {%
2792 \protected@edef\theHglentrycounter{\expandafter\noexpand
2793 \csname theH\@gls@counter\endcsname}%
2794 }%
2795 \fi
2796 }

```



`\@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.

```

2797 \def\@set@glo@numformat#1#2#3#4{%
2798   \expandafter\@glo@check@mkidxrangechar#3\@nil
2799   \protected@edef#1{%
2800     \@glo@prefix setentrycounter[#4]{#2}%
2801     \expandafter\string\csname\@glo@suffix\endcsname
2802   }%
2803   \@gls@checkmkidxchars#1%
2804 }

```

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.

```

2805 \def\@glo@check@mkidxrangechar#1#2\@nil{%
2806   \if#1(\relax
2807     \def\@glo@prefix{}%
2808     \if\relax#2\relax
2809       \def\@glo@suffix{glsnumberformat}%
2810     \else
2811       \def\@glo@suffix{#2}%
2812     \fi
2813   \else
2814     \if#1)\relax
2815       \def\@glo@prefix{)}%
2816     \if\relax#2\relax
2817       \def\@glo@suffix{glsnumberformat}%
2818     \else
2819       \def\@glo@suffix{#2}%
2820     \fi
2821   \else
2822     \def\@glo@prefix{}\def\@glo@suffix{#1#2}%
2823   \fi
2824 \fi}

```

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

```

2825 \newcommand*\@gls@escbsdq[1]{%
2826   \def\@gls@checkedmkidx{%
2827     \let\gls@xdystring=#1\relax
2828     \@onelevel@sanitize\gls@xdystring
2829     \edef\do@gls@xdycheckbackslash{%
2830       \noexpand\@gls@xdycheckbackslash\gls@xdystring\noexpand\@nil
2831       \@backslashchar\@backslashchar\noexpand\null}%

```

```

2832 \do@glx@xdycheckbackslash
2833 \expandafter\@glx@updatechecked\@glx@checkedmkidx{\glx@xdystring}%
2834 \def\@glx@checkedmkidx{%
2835 \expandafter\@glx@xdycheckquote\glx@xdystring\@nil""\null
2836 \expandafter\@glx@updatechecked\@glx@checkedmkidx{\glx@xdystring}%

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

2837 \@for\@glx@tmp:=\glx@protected@pagefmts\do
2838 {%
2839 \edef\@glx@sanitized@tmp{\expandafter\@gobble\string\\expandonce\@glx@tmp}%
2840 \@onelevel@sanitize\@glx@sanitized@tmp
2841 \edef\glx@dosubst{%
2842 \noexpand\DTLsubstituteall\noexpand\glx@xdystring
2843 {\@glx@sanitized@tmp}{\expandonce\@glx@tmp}%
2844 }%
2845 \glx@dosubst
2846 }%

Assign to required control sequence
2847 \let#1=\glx@xdystring
2848 }

```

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

glx@checkmkidxchars

```

2849 \newcommand{\@glx@checkmkidxchars}[1]{%
2850 \ifglxindy
2851 \@glx@escbsdq{#1}%
2852 \else
2853 \def\@glx@checkedmkidx{%
2854 \expandafter\@glx@checkquote#1\@nil""\null
2855 \expandafter\@glx@updatechecked\@glx@checkedmkidx{#1}%
2856 \def\@glx@checkedmkidx{%
2857 \expandafter\@glx@checkescquote#1\@nil\""\null
2858 \expandafter\@glx@updatechecked\@glx@checkedmkidx{#1}%
2859 \def\@glx@checkedmkidx{%
2860 \expandafter\@glx@checkescactual#1\@nil\?\?\null
2861 \expandafter\@glx@updatechecked\@glx@checkedmkidx{#1}%
2862 \def\@glx@checkedmkidx{%
2863 \expandafter\@glx@checkactual#1\@nil??\null
2864 \expandafter\@glx@updatechecked\@glx@checkedmkidx{#1}%
2865 \def\@glx@checkedmkidx{%
2866 \expandafter\@glx@checkbar#1\@nil||\null
2867 \expandafter\@glx@updatechecked\@glx@checkedmkidx{#1}%
2868 \def\@glx@checkedmkidx{%
2869 \expandafter\@glx@checkescbar#1\@nil\\|\null
2870 \expandafter\@glx@updatechecked\@glx@checkedmkidx{#1}%
2871 \def\@glx@checkedmkidx{%
2872 \expandafter\@glx@checklevel#1\@nil!!\null

```

```

2873 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2874 \fi
2875 }

```

Update the control sequence and strip trailing \@nil:

\@gls@updatechecked

```

2876 \def\@gls@updatechecked#1\@nil#2{\def#2{#1}}

```

\@gls@tmpb Define temporary token

```

2877 \newtoks\@gls@tmpb

```

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

```

2878 \def\@gls@checkquote#1"#2"#3\null{%
2879 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2880 \toks@={#1}%
2881 \ifx\null#2\null
2882 \ifx\null#3\null
2883 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2884 \def\@gls@checkquote{\relax}%
2885 \else
2886 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2887 \@gls@quotechar\@gls@quotechar\@gls@quotechar\@gls@quotechar}%
2888 \def\@gls@checkquote{\@gls@checkquote#3\null}%
2889 \fi
2890 \else
2891 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2892 \@gls@quotechar\@gls@quotechar}%
2893 \ifx\null#3\null
2894 \def\@gls@checkquote{\@gls@checkquote#2""\null}%
2895 \else
2896 \def\@gls@checkquote{\@gls@checkquote#2"#3\null}%
2897 \fi
2898 \fi
2899 \@gls@checkquote
2900 }

```

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

```

2901 \def\@gls@checkescquote#1\"#2\"#3\null{%
2902 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2903 \toks@={#1}%
2904 \ifx\null#2\null
2905 \ifx\null#3\null
2906 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2907 \def\@gls@checkescquote{\relax}%
2908 \else
2909 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2910 \@gls@quotechar\string\" \@gls@quotechar
2911 \@gls@quotechar\string\" \@gls@quotechar}%

```

```

2912 \def\@gls@checkescquote{\@gls@checkescquote#3\null}%
2913 \fi
2914 \else
2915 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2916 \@gls@quotearchar\string"\@gls@quotearchar}%
2917 \ifx\null#3\null
2918 \def\@gls@checkescquote{\@gls@checkescquote#2\""\null}%
2919 \else
2920 \def\@gls@checkescquote{\@gls@checkescquote#2\">#3\null}%
2921 \fi
2922 \fi
2923 \@gls@checkescquote
2924 }

```

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

```

2925 \def\@gls@checkescactual#1\?#2\?#3\null{%
2926 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2927 \toks@={#1}%
2928 \ifx\null#2\null
2929 \ifx\null#3\null
2930 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2931 \def\@gls@checkescactual{\relax}%
2932 \else
2933 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2934 \@gls@quotearchar\string"\@gls@actualchar
2935 \@gls@quotearchar\string"\@gls@actualchar}%
2936 \def\@gls@checkescactual{\@gls@checkescactual#3\null}%
2937 \fi
2938 \else
2939 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2940 \@gls@quotearchar\string"\@gls@actualchar}%
2941 \ifx\null#3\null
2942 \def\@gls@checkescactual{\@gls@checkescactual#2\?\?\null}%
2943 \else
2944 \def\@gls@checkescactual{\@gls@checkescactual#2\?#3\null}%
2945 \fi
2946 \fi
2947 \@gls@checkescactual
2948 }

```

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

```

2949 \def\@gls@checkescbar#1\|#2\|#3\null{%
2950 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2951 \toks@={#1}%
2952 \ifx\null#2\null
2953 \ifx\null#3\null
2954 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2955 \def\@gls@checkescbar{\relax}%
2956 \else

```

```

2957 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2958 \@gls@quotearchar\string\"@gls@encapchar
2959 \@gls@quotearchar\string\"@gls@encapchar}%
2960 \def\@@gls@checkescbar{\@gls@checkescbar#3\null}%
2961 \fi
2962 \else
2963 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2964 \@gls@quotearchar\string\"@gls@encapchar}%
2965 \ifx\null#3\null
2966 \def\@@gls@checkescbar{\@gls@checkescbar#2\|\|\null}%
2967 \else
2968 \def\@@gls@checkescbar{\@gls@checkescbar#2\|#3\null}%
2969 \fi
2970 \fi
2971 \@@gls@checkescbar
2972 }

```

\@gls@checkesclevel Similarly for \!:

```

2973 \def\@gls@checkesclevel#1\!#2\!#3\null{%
2974 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2975 \toks@={#1}%
2976 \ifx\null#2\null
2977 \ifx\null#3\null
2978 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2979 \def\@@gls@checkesclevel{\relax}%
2980 \else
2981 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2982 \@gls@quotearchar\string\"@gls@levelchar
2983 \@gls@quotearchar\string\"@gls@levelchar}%
2984 \def\@@gls@checkesclevel{\@gls@checkesclevel#3\null}%
2985 \fi
2986 \else
2987 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2988 \@gls@quotearchar\string\"@gls@levelchar}%
2989 \ifx\null#3\null
2990 \def\@@gls@checkesclevel{\@gls@checkesclevel#2\!\!\null}%
2991 \else
2992 \def\@@gls@checkesclevel{\@gls@checkesclevel#2\!#3\null}%
2993 \fi
2994 \fi
2995 \@@gls@checkesclevel
2996 }

```

\@gls@checkbar and for |:

```

2997 \def\@gls@checkbar#1|#2|#3\null{%
2998 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2999 \toks@={#1}%
3000 \ifx\null#2\null
3001 \ifx\null#3\null

```

```

3002 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
3003 \def\@@gls@checkbar{\relax}%
3004 \else
3005 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3006 \@gls@quotechar\@gls@encapchar\@gls@quotechar\@gls@encapchar}%
3007 \def\@@gls@checkbar{\@gls@checkbar#3\null}%
3008 \fi
3009 \else
3010 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3011 \@gls@quotechar\@gls@encapchar}%
3012 \ifx\null#3\null
3013 \def\@@gls@checkbar{\@gls@checkbar#2||\null}%
3014 \else
3015 \def\@@gls@checkbar{\@gls@checkbar#2|#3\null}%
3016 \fi
3017 \fi
3018 \@@gls@checkbar
3019 }

```

\@gls@checklevel and for !:

```

3020 \def\@gls@checklevel#1!#2!#3\null{%
3021 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
3022 \toks@={#1}%
3023 \ifx\null#2\null
3024 \ifx\null#3\null
3025 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
3026 \def\@@gls@checklevel{\relax}%
3027 \else
3028 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3029 \@gls@quotechar\@gls@levelchar\@gls@quotechar\@gls@levelchar}%
3030 \def\@@gls@checklevel{\@gls@checklevel#3\null}%
3031 \fi
3032 \else
3033 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3034 \@gls@quotechar\@gls@levelchar}%
3035 \ifx\null#3\null
3036 \def\@@gls@checklevel{\@gls@checklevel#2!!\null}%
3037 \else
3038 \def\@@gls@checklevel{\@gls@checklevel#2!#3\null}%
3039 \fi
3040 \fi
3041 \@@gls@checklevel
3042 }

```

\@gls@checkactual and for ?:

```

3043 \def\@gls@checkactual#1?#2?#3\null{%
3044 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
3045 \toks@={#1}%
3046 \ifx\null#2\null

```

```

3047 \ifx\null#3\null
3048 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
3049 \def\@@gls@checkactual{\relax}%
3050 \else
3051 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3052 \@gls@quotechar\@gls@actualchar\@gls@quotechar\@gls@actualchar}%
3053 \def\@@gls@checkactual{\@gls@checkactual#3\null}%
3054 \fi
3055 \else
3056 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3057 \@gls@quotechar\@gls@actualchar}%
3058 \ifx\null#3\null
3059 \def\@@gls@checkactual{\@gls@checkactual#2??\null}%
3060 \else
3061 \def\@@gls@checkactual{\@gls@checkactual#2?#3\null}%
3062 \fi
3063 \fi
3064 \@@gls@checkactual
3065 }

```

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

```

3066 \def\@gls@xdycheckquote#1"#2"#3\null{%
3067 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
3068 \toks@={#1}%
3069 \ifx\null#2\null
3070 \ifx\null#3\null
3071 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
3072 \def\@@gls@xdycheckquote{\relax}%
3073 \else
3074 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3075 \string\string\}%
3076 \def\@@gls@xdycheckquote{\@gls@xdycheckquote#3\null}%
3077 \fi
3078 \else
3079 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3080 \string\}%
3081 \ifx\null#3\null
3082 \def\@@gls@xdycheckquote{\@gls@xdycheckquote#2""\null}%
3083 \else
3084 \def\@@gls@xdycheckquote{\@gls@xdycheckquote#2"#3\null}%
3085 \fi
3086 \fi
3087 \@@gls@xdycheckquote
3088 }

```

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

```

\@gls@xdycheckbackslash
3089 \edef\def\@gls@xdycheckbackslash{%
3090 \noexpand\def\noexpand\@gls@xdycheckbackslash##1\@backslashchar

```

```

3091    ##2\@backslashchar##3\@noexpand\null{%
3092 \noexpand\@gls@tmpb=\noexpand\expandafter
3093   {\noexpand\@gls@checkedmkidx}%
3094 \noexpand\toks@={##1}%
3095 \noexpand\ifx\noexpand\null##2\noexpand\null
3096 \noexpand\ifx\noexpand\null##3\noexpand\null
3097 \noexpand\edef\noexpand\@gls@checkedmkidx{%
3098   \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@}%
3099 \noexpand\def\noexpand\@gls@xdycheckbackslash{\relax}%
3100 \noexpand\else
3101 \noexpand\edef\noexpand\@gls@checkedmkidx{%
3102   \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
3103   \@backslashchar\@backslashchar\@backslashchar\@backslashchar}%
3104 \noexpand\def\noexpand\@gls@xdycheckbackslash{%
3105   \noexpand\@gls@xdycheckbackslash##3\noexpand\null}%
3106 \noexpand\fi
3107 \noexpand\else
3108 \noexpand\edef\noexpand\@gls@checkedmkidx{%
3109   \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
3110   \@backslashchar\@backslashchar}%
3111 \noexpand\ifx\noexpand\null##3\noexpand\null
3112 \noexpand\def\noexpand\@gls@xdycheckbackslash{%
3113   \noexpand\@gls@xdycheckbackslash##2\@backslashchar
3114   \@backslashchar\noexpand\null}%
3115 \noexpand\else
3116 \noexpand\def\noexpand\@gls@xdycheckbackslash{%
3117   \noexpand\@gls@xdycheckbackslash##2\@backslashchar
3118   ##3\noexpand\null}%
3119 \noexpand\fi
3120 \noexpand\fi
3121 \noexpand\@gls@xdycheckbackslash
3122 }%
3123 }

```

Now go ahead and define \@gls@xdycheckbackslash

```

3124 \def@gls@xdycheckbackslash

```

\glsdohypertarget

```

3125 \newlength\gls@tmplen
3126 \newcommand*\glsdohypertarget[2]{%
3127   \settoheight{\gls@tmplen}{#2}%
3128   \raisebox{\gls@tmplen}{\hypertarget{#1}{}}#2%
3129 }

```

\glsdohyperlink

```

3130 \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.



```

3131 \ifcsundef{hyperlink}%
3132 {%
3133   \let\@glslink\@secondoftwo
3134 }%
3135 {%
3136   \let\@glslink\glsdohyperlink
3137 }

```

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

```

3138 \ifcsundef{hypertarget}%
3139 {%
3140   \let\@glstarget\@secondoftwo
3141 }%
3142 {%
3143   \let\@glstarget\glsdohypertarget
3144 }

```

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

`\glsdisablehyper`

```

3145 \newcommand{\glsdisablehyper}{%
3146   \KV@glslink@hyperfalse
3147   \let\@glslink\@secondoftwo
3148   \let\@glstarget\@secondoftwo
3149 }

```

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

`\glsenablehyper`

```

3150 \newcommand{\glsenablehyper}{%
3151   \KV@glslink@hypertrue
3152   \let\@glslink\glsdohyperlink
3153   \let\@glstarget\glsdohypertarget
3154 }

```

Provide some convenience commands if not already defined:

```

3155 \providecommand{\@firstofthree}[3]{#1}
3156 \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
3157 \newrobustcmd*{\gls}{\@gls@hyp@opt\@gls}
```

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

```
\@gls
3158 \newcommand*{\@gls}[2] [] {%
3159   \new@ifnextchar[\@gls@{#1}{#2}]{\@gls@{#1}{#2} []}%
3160 }
```

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

```
3161 \def\@gls@#1#2[#3]{%
3162   \glsdoifexists{#2}%
3163   {%
3164     \let\do@gls@link@checkfirsthyper\@gls@link@checkfirsthyper

3165     \let\glsifplural\@secondoftwo
3166     \let\glsupcase\@firstofthree
3167     \let\glscustomtext\@empty
3168     \def\glsinsert{#3}%
```

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

```
3169   \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.

```
3170   \@gls@link[#1]{#2}{\@glo@text}%
```

Indicate that this entry has now been used

```
3171   \ifKV@glslink@local
3172     \glslocalunset{#2}%
3173   \else
3174     \glsunset{#2}%
3175   \fi
3176 }%
3177 }
```

`\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`

```
3178 \newrobustcmd*{\Gls}{\@gls@hyp@opt\@Gls}
```

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

```
3179 \newcommand*{\@Gls}[2] [] {%
3180   \new@ifnextchar[{\@Gls@{#1}{#2}}{\@Gls@{#1}{#2} []}%
3181 }
```

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

```
3182 \def\@Gls@#1#2[#3]{%
3183   \glsdoifexists{#2}%
3184   {%
3185     \let\do@gls@link@checkfirsthyper\@gls@link@checkfirsthyper

3186     \let\glsifplural\@secondoftwo
3187     \let\glscapscase\@secondofthree
3188     \let\glscustomtext\@empty
3189     \def\glsinsert{#3}%
```

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

```
3190   \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.

```
3191   \@gls@link[#1]{#2}{\@glo@text}%
```

Indicate that this entry has now been used

```
3192   \ifKV@gls@link@local
3193     \glslocalunset{#2}%
3194   \else
3195     \glsunset{#2}%
3196   \fi
3197 }%
3198 }
```

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

`\GLS`

```
3199 \newrobustcmd*{\GLS}{\@gls@hyp@opt\@GLS}
```

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

```
3200 \newcommand*{\@GLS}[2] [] {%
3201   \new@ifnextchar[{\@GLS@{#1}{#2}}{\@GLS@{#1}{#2} []}%
3202 }
```

\@GLS@ Read in the final optional argument:

```
3203 \def\@GLS@#1#2[#3] {%
3204   \glsdoifexists{#2}%
3205   {%
3206     \let\do@glsl@link@checkfirsthyper\@glsl@link@checkfirsthyper
3207     \let\glsl@link@secondoftwo
3208     \let\glsl@link@thirdofthree
3209     \let\glsl@link@empty
3210     \def\glsl@link{#3}%
3211     \def\@glo@text{\csname gls@\glstype @entryfmt\endcsname}%
3212     Call \@glsl@link If footnote package option has been used and the glossary
3213     type is \acronymtype, suppress hyperlink for first use. Likewise if the hyper-
3214     first=false package option is used.
3215     \@glsl@link[#1]{#2}{\@glo@text}%
3216     Indicate that this entry has now been used
3217     \ifKV@glsl@link@local
3218       \glsl@link@localunset{#2}%
3219     \else
3220       \glsl@link@unset{#2}%
3221     \fi
3222   }%
3223 }
```

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

```
3211 \def\@glo@text{\csname gls@\glstype @entryfmt\endcsname}%
```

Call \@glsl@link If footnote package option has been used and the glossary type is \acronymtype, suppress hyperlink for first use. Likewise if the hyper-first=false package option is used.

```
3212 \@glsl@link[#1]{#2}{\@glo@text}%
```

Indicate that this entry has now been used

```
3213 \ifKV@glsl@link@local
3214   \glsl@link@localunset{#2}%
3215 \else
3216   \glsl@link@unset{#2}%
3217 \fi
3218 }%
3219 }
```

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

\glspl

```
3220 \newrobustcmd*{\glspl}{\@glsl@hyp@opt\@glspl}
```

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

```
3221 \newcommand*{\@glspl}[2] [] {%
3222   \new@ifnextchar[{\@glspl@{#1}{#2}}{\@glspl@{#1}{#2} []}%
3223 }
```

\@glspl@ Read in the final optional argument:

```
3224 \def\@glspl@#1#2[#3] {%
3225   \glsdoifexists{#2}%
3226   {%
3227     \let\do@glsl@link@checkfirsthyper\@glsl@link@checkfirsthyper
3228     \let\glsl@link@secondoftwo
3229     \let\glsl@link@thirdofthree
3230     \let\glsl@link@empty
3231     \def\glsl@link{#3}%
3232     \def\@glo@text{\csname gls@\glstype @entryfmt\endcsname}%
3233     Call \@glspl@link If footnote package option has been used and the glossary
3234     type is \acronymtype, suppress hyperlink for first use. Likewise if the hyper-
3235     first=false package option is used.
3236     \@glspl@link[#1]{#2}{\@glo@text}%
3237     Indicate that this entry has now been used
3238     \ifKV@glspl@link@local
3239       \glspl@link@localunset{#2}%
3240     \else
3241       \glspl@link@unset{#2}%
3242     \fi
3243   }%
3244 }
```

```

3228 \let\glsifplural\@firstoftwo
3229 \let\glscapscase\@firstofthree
3230 \let\glscustomtext\@empty
3231 \def\glsinsert{#3}%

```

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

```

3232 \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.

```

3233 \@gls@link[#1]{#2}{\@glo@text}%

```

Indicate that this entry has now been used

```

3234 \ifKV@glslink@local
3235 \glslocalunset{#2}%
3236 \else
3237 \glsunset{#2}%
3238 \fi
3239 }%
3240 }

```

\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

```

3241 \newrobustcmd*{\Glspl}{\@gls@hyp@opt\@Glspl}

```

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

```

3242 \newcommand*{\@Glspl}[2][]{%
3243 \new@ifnextchar[\@Glspl@{#1}{#2}]{\@Glspl@{#1}{#2}[]}%
3244 }

```

\@Glspl@ Read in the final optional argument:

```

3245 \def\@Glspl@#1#2[#3]{%
3246 \glsdoifexists{#2}%
3247 {%
3248 \let\do@gls@link@checkfirsthyper\@gls@link@checkfirsthyper

3249 \let\glsifplural\@firstoftwo
3250 \let\glscapscase\@secondofthree
3251 \let\glscustomtext\@empty
3252 \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.

```

3253 \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.

```
3254 \gls@link[#1]{#2}{\@glo@text}%
```

Indicate that this entry has now been used

```
3255 \ifKV@glslink@local
3256 \glslocalunset{#2}%
3257 \else
3258 \glsunset{#2}%
3259 \fi
3260 }%
3261 }
```

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

`\GLSp1`

```
3262 \newrobustcmd*{\GLSp1}{\@gls@hyp@opt\@GLSp1}
```

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

```
3263 \newcommand*{\@GLSp1}[2][{}]{%
3264 \new@ifnextchar[{\@GLSp1@{#1}{#2}}{\@GLSp1@{#1}{#2}[]}%
3265 }
```

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

```
3266 \def\@GLSp1@#1#2[#3]{%
3267 \glsdoifexists{#2}%
3268 {%
3269 \let\do@gls@link@checkfirsthyper\@gls@link@checkfirsthyper

3270 \let\glsifplural\@firstoftwo
3271 \let\glscapscase\@thirdofthree
3272 \let\glscustomtext\@empty
3273 \def\glsinsert{#3}%
```

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

```
3274 \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.

```
3275 \gls@link[#1]{#2}{\@glo@text}%
```

Indicate that this entry has now been used

```
3276 \ifKV@glslink@local
3277 \glslocalunset{#2}%
3278 \else
```

```

3279     \glsunset{#2}%
3280   \fi
3281 }%
3282 }

```

`\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:

```

3283 \newrobustcmd*{\glsdisp}{\@gls@hyp@opt\@glsdisp}

```

Defined the un-starred form.

`\@glsdisp`

```

3284 \newcommand*{\@glsdisp}[3][\%
3285   \glsdoifexists{#2}]{%

3286   \let\do@gls@link@checkfirsthyper\@gls@link@checkfirsthyper

3287   \let\glsifplural\@secondoftwo
3288   \let\glsupcase\@firstofthree
3289   \def\glscustomtext{#3}%
3290   \def\glsinsert{}}%

```

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

```

3291   \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.

```

3292   \@gls@link[#1]{#2}{\@glo@text}%

```

Indicate that this entry has now been used

```

3293   \ifKV@glslink@local
3294     \glslocalunset{#2}%
3295   \else
3296     \glsunset{#2}%
3297   \fi
3298 }%
3299 }

```

`\@gls@field@link`

```

3300 \newcommand{\@gls@field@link}[3]{%
3301   \glsdoifexists{#2}%
3302   {%
3303     \let\do@gls@link@checkfirsthyper\relax
3304     \@gls@link[#1]{#2}{#3}%
3305   }%
3306 }

```

`\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`

```
3307 \newrobustcmd*{\glstext}{\@gls@hyp@opt\@glstext}
```

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

```
3308 \newcommand*{\@glstext}[2][\%
```

```
3309 \new@ifnextchar[\@glstext@{#1}{#2}]{\@glstext@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3310 \def\@glstext@#1#2[#3]{\%
```

```
3311 \@gls@field@link{#1}{#2}{\glsentrytext{#2}#3}%
```

```
3312 }
```

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

`\GLStext`

```
3313 \newrobustcmd*{\GLStext}{\@gls@hyp@opt\@GLStext}
```

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

```
3314 \newcommand*{\@GLStext}[2][\%
```

```
3315 \new@ifnextchar[\@GLStext@{#1}{#2}]{\@GLStext@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3316 \def\@GLStext@#1#2[#3]{\%
```

```
3317 \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentrytext{#2}#3}}%
```

```
3318 }
```

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

`\Glstext`

```
3319 \newrobustcmd*{\Glstext}{\@gls@hyp@opt\@Glstext}
```

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

```
3320 \newcommand*{\@Glstext}[2][\%
```

```
3321 \new@ifnextchar[\@Glstext@{#1}{#2}]{\@Glstext@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3322 \def\@Glstext@#1#2[#3]{\%
```

```
3323 \@gls@field@link{#1}{#2}{\Glsentrytext{#2}#3}%
```

```
3324 }
```

`\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`

```
3325 \newrobustcmd*{\glsfirst}{\@gls@hyp@opt\@glsfirst}
```



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

```
3326 \newcommand*{\@glsfirst}[2] [] {%
3327   \new@ifnextchar[{\@glsfirst@{#1}{#2}}]{\@glsfirst@{#1}{#2} []}}
```

Read in the final optional argument:

```
3328 \def\@glsfirst@#1#2[#3] {%
3329   \@gls@field@link{#1}{#2}{\glsentryfirst{#2}#3}%
3330 }
```

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

\Glsfirst

```
3331 \newrobustcmd*{\Glsfirst}{\@gls@hyp@opt\@Glsfirst}
```

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

```
3332 \newcommand*{\@Glsfirst}[2] [] {%
3333   \new@ifnextchar[{\@Glsfirst@{#1}{#2}}]{\@Glsfirst@{#1}{#2} []}}
```

Read in the final optional argument:

```
3334 \def\@Glsfirst@#1#2[#3] {%
3335   \@gls@field@link{#1}{#2}{\Glsentryfirst{#2}#3}%
3336 }
```

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

\GLSfirst

```
3337 \newrobustcmd*{\GLSfirst}{\@gls@hyp@opt\@GLSfirst}
```

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

```
3338 \newcommand*{\@GLSfirst}[2] [] {%
3339   \new@ifnextchar[{\@GLSfirst@{#1}{#2}}]{\@GLSfirst@{#1}{#2} []}}
```

Read in the final optional argument:

```
3340 \def\@GLSfirst@#1#2[#3] {%
3341   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryfirst{#2}#3}}%
3342 }
```

\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

```
3343 \newrobustcmd*{\glsplural}{\@gls@hyp@opt\@glsplural}
```

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

```
3344 \newcommand*{\@glsplural}[2] [] {%
3345   \new@ifnextchar[{\@glsplural@{#1}{#2}}]{\@glsplural@{#1}{#2} []}}
```

Read in the final optional argument:

```
3346 \def\@glsplural@#1#2[#3]{%
3347   \@gls@field@link{#1}{#2}{\glsentryplural{#2}#3}%
3348 }
```

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

\Glsplural

```
3349 \newrobustcmd*{\Glsplural}{\@gls@hyp@opt\@Glsplural}
```

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

```
3350 \newcommand*{\@Glsplural}[2][\%
3351   \new@ifnextchar[{\@Glsplural@{#1}{#2}}{\@Glsplural@{#1}{#2}}{ }]
```

Read in the final optional argument:

```
3352 \def\@Glsplural@#1#2[#3]{%
3353   \@gls@field@link{#1}{#2}{\glsentryplural{#2}#3}%
3354 }
```

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

\GLSplural

```
3355 \newrobustcmd*{\GLSplural}{\@gls@hyp@opt\@GLSplural}
```

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

```
3356 \newcommand*{\@GLSplural}[2][\%
3357   \new@ifnextchar[{\@GLSplural@{#1}{#2}}{\@GLSplural@{#1}{#2}}{ }]
```

Read in the final optional argument:

```
3358 \def\@GLSplural@#1#2[#3]{%
3359   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryplural{#2}#3}}%
3360 }
```

\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

```
3361 \newrobustcmd*{\glsfirstplural}{\@gls@hyp@opt\@glsfirstplural}
```

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

```
3362 \newcommand*{\@glsfirstplural}[2][\%
3363   \new@ifnextchar[{\@glsfirstplural@{#1}{#2}}{\@glsfirstplural@{#1}{#2}}{ }]
```

Read in the final optional argument:

```
3364 \def\@glsfirstplural@#1#2[#3]{%
3365   \@gls@field@link{#1}{#2}{\glsentryfirstplural{#2}#3}%
3366 }
```

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

`\Glsfirstplural`

```
3367 \newrobustcmd*{\Glsfirstplural}{\@gls@hyp@opt\@Glsfirstplural}
```

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

```
3368 \newcommand*{\@Glsfirstplural}[2][\@Glsfirstplural@{#1}{#2}[]]{%
```

```
3369 \new@ifnextchar[\@Glsfirstplural@{#1}{#2}]{\@Glsfirstplural@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3370 \def\@Glsfirstplural@#1#2[#3]{%
```

```
3371 \@gls@field@link{#1}{#2}{\Glsentryfirstplural{#2}#3}%
```

```
3372 }
```

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

`\GLSfirstplural`

```
3373 \newrobustcmd*{\GLSfirstplural}{\@gls@hyp@opt\@GLSfirstplural}
```

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

```
3374 \newcommand*{\@GLSfirstplural}[2][\@GLSfirstplural@{#1}{#2}[]]{%
```

```
3375 \new@ifnextchar[\@GLSfirstplural@{#1}{#2}]{\@GLSfirstplural@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3376 \def\@GLSfirstplural@#1#2[#3]{%
```

```
3377 \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\Glsentryfirstplural{#2}#3}}%
```

```
3378 }
```

`\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`

```
3379 \newrobustcmd*{\glsname}{\@gls@hyp@opt\@glsname}
```

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

```
3380 \newcommand*{\@glsname}[2][\@glsname@{#1}{#2}[]]{%
```

```
3381 \new@ifnextchar[\@glsname@{#1}{#2}]{\@glsname@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3382 \def\@glsname@#1#2[#3]{%
```

```
3383 \@gls@field@link{#1}{#2}{\Glsentryname{#2}#3}%
```

```
3384 }
```

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

`\Glsname`

```
3385 \newrobustcmd*{\Glsname}{\@gls@hyp@opt\@Glsname}
```

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

```
3386 \newcommand*{\@Glsname}[2] [] {%
3387   \new@ifnextchar[{\@Glsname@{#1}{#2}}{\@Glsname@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3388 \def\@Glsname@#1#2[#3] {%
3389   \@gls@field@link{#1}{#2}{\Glsentryname{#2}#3}%
3390 }
```

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

\GLSname

```
3391 \newrobustcmd*{\GLSname}{\@gls@hyp@opt\@GLSname}
```

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

```
3392 \newcommand*{\@GLSname}[2] [] {%
3393   \new@ifnextchar[{\@GLSname@{#1}{#2}}{\@GLSname@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3394 \def\@GLSname@#1#2[#3] {%
3395   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\Glsentryname{#2}#3}}%
3396 }
```

\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

```
3397 \newrobustcmd*{\glsdesc}{\@gls@hyp@opt\@glsdesc}
```

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

```
3398 \newcommand*{\@glsdesc}[2] [] {%
3399   \new@ifnextchar[{\@glsdesc@{#1}{#2}}{\@glsdesc@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3400 \def\@glsdesc@#1#2[#3] {%
3401   \@gls@field@link{#1}{#2}{\Glsentrydesc{#2}#3}%
3402 }
```

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

\Glsdesc

```
3403 \newrobustcmd*{\Glsdesc}{\@gls@hyp@opt\@Glsdesc}
```

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

```
3404 \newcommand*{\@Glsdesc}[2] [] {%
3405   \new@ifnextchar[{\@Glsdesc@{#1}{#2}}{\@Glsdesc@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3406 \def\@GLSdesc@#1#2[#3]{%
3407   \@gls@field@link{#1}{#2}{\Glsentrydesc{#2}#3}%
3408 }
```

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

\GLSdesc

```
3409 \newrobustcmd*{\GLSdesc}{\@gls@hyp@opt\@GLSdesc}
```

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

```
3410 \newcommand*{\@GLSdesc}[2][ ]{%
3411   \new@ifnextchar[{\@GLSdesc@{#1}{#2}}{\@GLSdesc@{#1}{#2}[ ]}]}
```

Read in the final optional argument:

```
3412 \def\@GLSdesc@#1#2[#3]{%
3413   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\Glsentrydesc{#2}#3}}%
3414 }
```

\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

```
3415 \newrobustcmd*{\glsdescplural}{\@gls@hyp@opt\@glsdescplural}
```

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

```
3416 \newcommand*{\@glsdescplural}[2][ ]{%
3417   \new@ifnextchar[{\@glsdescplural@{#1}{#2}}{\@glsdescplural@{#1}{#2}[ ]}]}
```

Read in the final optional argument:

```
3418 \def\@glsdescplural@#1#2[#3]{%
3419   \@gls@field@link{#1}{#2}{\Glsentrydescplural{#2}#3}%
3420 }
```

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

\Glsdescplural

```
3421 \newrobustcmd*{\Glsdescplural}{\@gls@hyp@opt\@Glsdescplural}
```

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

```
3422 \newcommand*{\@Glsdescplural}[2][ ]{%
3423   \new@ifnextchar[{\@Glsdescplural@{#1}{#2}}{\@Glsdescplural@{#1}{#2}[ ]}]}
```

Read in the final optional argument:

```
3424 \def\@Glsdescplural@#1#2[#3]{%
3425   \@gls@field@link{#1}{#2}{\Glsentrydescplural{#2}#3}%
3426 }
```

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

`\GLSdescplural`

```
3427 \newrobustcmd*{\GLSdescplural}{\@gls@hyp@opt\@GLSdescplural}
```

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

```
3428 \newcommand*{\@GLSdescplural}[2] [] {%
```

```
3429   \new@ifnextchar[{\@GLSdescplural@{#1}{#2}}{\@GLSdescplural@{#1}{#2} []}]
```

Read in the final optional argument:

```
3430 \def\@GLSdescplural@#1#2[#3] {%
```

```
3431   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentrydescplural{#2}{#3}}}%
```

```
3432 }
```

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

`\glsymbol`

```
3433 \newrobustcmd*{\glsymbol}{\@gls@hyp@opt\@glsymbol}
```

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

```
3434 \newcommand*{\@glsymbol}[2] [] {%
```

```
3435   \new@ifnextchar[{\@glsymbol@{#1}{#2}}{\@glsymbol@{#1}{#2} []}]
```

Read in the final optional argument:

```
3436 \def\@glsymbol@#1#2[#3] {%
```

```
3437   \@gls@field@link{#1}{#2}{\glsentrysymbol{#2}{#3}}%
```

```
3438 }
```

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

`\Glsymbol`

```
3439 \newrobustcmd*{\Glsymbol}{\@gls@hyp@opt\@Glsymbol}
```

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

```
3440 \newcommand*{\@Glsymbol}[2] [] {%
```

```
3441   \new@ifnextchar[{\@Glsymbol@{#1}{#2}}{\@Glsymbol@{#1}{#2} []}]
```

Read in the final optional argument:

```
3442 \def\@Glsymbol@#1#2[#3] {%
```

```
3443   \@gls@field@link{#1}{#2}{\Glsentrysymbol{#2}{#3}}%
```

```
3444 }
```

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

`\GLSsymbol`

```
3445 \newrobustcmd*{\GLSsymbol}{\@gls@hyp@opt\@GLSsymbol}
```

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

```
3446 \newcommand*{\@GLSsymbol}[2] [] {%
3447   \new@ifnextchar[{\@GLSsymbol@{#1}{#2}}{\@GLSsymbol@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3448 \def\@GLSsymbol@#1#2[#3] {%
3449   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentrysymbol{#2}#3}}%
3450 }
```

`\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`

```
3451 \newrobustcmd*{\glsymbolplural}{\@gls@hyp@opt\@glsymbolplural}
```

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

```
3452 \newcommand*{\@glsymbolplural}[2] [] {%
3453   \new@ifnextchar[{\@glsymbolplural@{#1}{#2}}{\@glsymbolplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3454 \def\@glsymbolplural@#1#2[#3] {%
3455   \@gls@field@link{#1}{#2}{\glsentrysymbolplural{#2}#3}%
3456 }
```

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

`\Glsymbolplural`

```
3457 \newrobustcmd*{\Glsymbolplural}{\@gls@hyp@opt\@Glsymbolplural}
```

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

```
3458 \newcommand*{\@Glsymbolplural}[2] [] {%
3459   \new@ifnextchar[{\@Glsymbolplural@{#1}{#2}}{\@Glsymbolplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3460 \def\@Glsymbolplural@#1#2[#3] {%
3461   \@gls@field@link{#1}{#2}{\Glsentrysymbolplural{#2}#3}%
3462 }
```

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

`\GLSsymbolplural`

```
3463 \newrobustcmd*{\GLSsymbolplural}{\@gls@hyp@opt\@GLSsymbolplural}
```

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

```
3464 \newcommand*{\@GLSsymbolplural}[2] [] {%
3465   \new@ifnextchar[{\@GLSsymbolplural@{#1}{#2}}{\@GLSsymbolplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3466 \def\@GLSsymbolplural@#1#2[#3]{%
3467   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentrysymbolplural{#2}#3}}%
3468 }
```

`\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`

```
3469 \newrobustcmd*{\glsuseri}{\@gls@hyp@opt\@glsuseri}
```

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

```
3470 \newcommand*{\@glsuseri}[2] []{%
3471   \new@ifnextchar[{\@glsuseri@{#1}{#2}}{\@glsuseri@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3472 \def\@glsuseri@#1#2[#3]{%
3473   \@gls@field@link{#1}{#2}{\glsentryuseri{#2}#3}%
3474 }
```

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

`\Glsuseri`

```
3475 \newrobustcmd*{\Glsuseri}{\@gls@hyp@opt\@Glsuseri}
```

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

```
3476 \newcommand*{\@Glsuseri}[2] []{%
3477   \new@ifnextchar[{\@Glsuseri@{#1}{#2}}{\@Glsuseri@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3478 \def\@Glsuseri@#1#2[#3]{%
3479   \@gls@field@link{#1}{#2}{\Glsentryuseri{#2}#3}%
3480 }
```

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

`\GLSuseri`

```
3481 \newrobustcmd*{\GLSuseri}{\@gls@hyp@opt\@GLSuseri}
```

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

```
3482 \newcommand*{\@GLSuseri}[2] []{%
3483   \new@ifnextchar[{\@GLSuseri@{#1}{#2}}{\@GLSuseri@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3484 \def\@GLSuseri@#1#2[#3]{%
3485   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryuseri{#2}#3}}%
3486 }
```



`\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`

```
3487 \newrobustcmd*{\glsuserii}{\@gls@hyp@opt\@glsuserii}
```

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

```
3488 \newcommand*{\@glsuserii}[2] [] {%
```

```
3489   \new@ifnextchar[{\@glsuserii@{#1}{#2}}{\@glsuserii@{#1}{#2} []}]
```

Read in the final optional argument:

```
3490 \def\@glsuserii@#1#2[#3] {%
```

```
3491   \@gls@field@link{#1}{#2}{\glsentryuserii{#2}#3}%
```

```
3492 }
```

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

`\Glsuserii`

```
3493 \newrobustcmd*{\Glsuserii}{\@gls@hyp@opt\@Glsuserii}
```

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

```
3494 \newcommand*{\@Glsuserii}[2] [] {%
```

```
3495   \new@ifnextchar[{\@Glsuserii@{#1}{#2}}{\@Glsuserii@{#1}{#2} []}]
```

Read in the final optional argument:

```
3496 \def\@Glsuserii@#1#2[#3] {%
```

```
3497   \@gls@field@link{#1}{#2}{\Glsentryuserii{#2}#3}%
```

```
3498 }
```

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

`\GLSuserii`

```
3499 \newrobustcmd*{\GLSuserii}{\@gls@hyp@opt\@GLSuserii}
```

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

```
3500 \newcommand*{\@GLSuserii}[2] [] {%
```

```
3501   \new@ifnextchar[{\@GLSuserii@{#1}{#2}}{\@GLSuserii@{#1}{#2} []}]
```

Read in the final optional argument:

```
3502 \def\@GLSuserii@#1#2[#3] {%
```

```
3503   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryuserii{#2}#3}}%
```

```
3504 }
```

`\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`

```
3505 \newrobustcmd*{\glsuseriii}{\@gls@hyp@opt\@glsuseriii}
```

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

```
3506 \newcommand*{\@glsuseriii}[2] [] {%
3507   \new@ifnextchar[{\@glsuseriii@{#1}{#2}}{\@glsuseriii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3508 \def\@glsuseriii@#1#2[#3]{%
3509   \@gls@field@link{#1}{#2}{\glstentryuseriii{#2}#3}%
3510 }
```

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

\Glsuseriii

```
3511 \newrobustcmd*{\Glsuseriii}{\@gls@hyp@opt\@Glsuseriii}
```

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

```
3512 \newcommand*{\@Glsuseriii}[2] [] {%
3513   \new@ifnextchar[{\@Glsuseriii@{#1}{#2}}{\@Glsuseriii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3514 \def\@Glsuseriii@#1#2[#3]{%
3515   \@gls@field@link{#1}{#2}{\GLstentryuseriii{#2}#3}%
3516 }
```

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

\GLSuseriii

```
3517 \newrobustcmd*{\GLSuseriii}{\@gls@hyp@opt\@GLSuseriii}
```

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

```
3518 \newcommand*{\@GLSuseriii}[2] [] {%
3519   \new@ifnextchar[{\@GLSuseriii@{#1}{#2}}{\@GLSuseriii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3520 \def\@GLSuseriii@#1#2[#3]{%
3521   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glstentryuseriii{#2}#3}}%
3522 }
```

\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

```
3523 \newrobustcmd*{\glsuseriv}{\@gls@hyp@opt\@glsuseriv}
```

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

```
3524 \newcommand*{\@glsuseriv}[2] [] {%
3525   \new@ifnextchar[{\@glsuseriv@{#1}{#2}}{\@glsuseriv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3526 \def\@glsuseriv@#1#2[#3]{%
3527   \@gls@field@link{#1}{#2}{\glsentryuseriv{#2}#3}%
3528 }
```

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

`\Glsuseriv`

```
3529 \newrobustcmd*{\Glsuseriv}{\@gls@hyp@opt\@Glsuseriv}
```

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

```
3530 \newcommand*{\@Glsuseriv}[2][ ]{%
3531   \new@ifnextchar[{\@Glsuseriv@{#1}{#2}}{\@Glsuseriv@{#1}{#2}[ ]}]}
```

Read in the final optional argument:

```
3532 \def\@Glsuseriv@#1#2[#3]{%
3533   \@gls@field@link{#1}{#2}{\glsentryuseriv{#2}#3}%
3534 }
```

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

`\GLSuseriv`

```
3535 \newrobustcmd*{\GLSuseriv}{\@gls@hyp@opt\@GLSuseriv}
```

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

```
3536 \newcommand*{\@GLSuseriv}[2][ ]{%
3537   \new@ifnextchar[{\@GLSuseriv@{#1}{#2}}{\@GLSuseriv@{#1}{#2}[ ]}]}
```

Read in the final optional argument:

```
3538 \def\@GLSuseriv@#1#2[#3]{%
3539   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryuseriv{#2}#3}}%
3540 }
```

`\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`

```
3541 \newrobustcmd*{\glsuserv}{\@gls@hyp@opt\@glsuserv}
```

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

```
3542 \newcommand*{\@glsuserv}[2][ ]{%
3543   \new@ifnextchar[{\@glsuserv@{#1}{#2}}{\@glsuserv@{#1}{#2}[ ]}]}
```

Read in the final optional argument:

```
3544 \def\@glsuserv@#1#2[#3]{%
3545   \@gls@field@link{#1}{#2}{\glsentryuserv{#2}#3}%
3546 }
```



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

```
3566 \newcommand*{\@Glsuservi}[2] [] {%
3567   \new@ifnextchar[{\@Glsuservi@{#1}{#2}}{\@Glsuservi@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3568 \def\@Glsuservi@#1#2[#3] {%
3569   \@gls@field@link{#1}{#2}{\glentryuservi{#2}#3}%
3570 }
```

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

\GLSuservi

```
3571 \newrobustcmd*{\GLSuservi}{\@gls@hyp@opt\@GLSuservi}
```

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

```
3572 \newcommand*{\@GLSuservi}[2] [] {%
3573   \new@ifnextchar[{\@GLSuservi@{#1}{#2}}{\@GLSuservi@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3574 \def\@GLSuservi@#1#2[#3] {%
3575   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glentryuservi{#2}#3}}%
3576 }
```

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

\acrshort

```
3577 \newrobustcmd*{\acrshort}{\@gls@hyp@opt\@ns@acrshort}
```

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

```
3578 \newcommand*{\@ns@acrshort}[2] [] {%
3579   \new@ifnextchar[{\@acrshort{#1}{#2}}{\@acrshort{#1}{#2} []}]%
3580 }
```

Read in the final optional argument:

```
3581 \def\@acrshort#1#2[#3] {%
3582   \glsdoifexists{#2}%
3583   {%
3584     \let\do@gls@link@checkfirsthyper\relax

3585     \let\glsifplural\@secondoftwo
3586     \let\glscapscase\@firstofthree
3587     \let\glsinsert\@empty
3588     \def\glscustomtext{%
3589       \acronymfont{\glentryshort{#2}}#3%
3590     }%
```

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

```
3591 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3592 }%
3593 }
```

\Acrshort

```
3594 \newrobustcmd*{\Acrshort}{\@gls@hyp@opt\@ns@Acrshort}
```

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

```
3595 \newcommand*{\ns@Acrshort}[2] [] {%
3596 \new@ifnextchar[{\@Acrshort{#1}{#2}}{\@Acrshort{#1}{#2} []}%
3597 }
```

Read in the final optional argument:

```
3598 \def\@Acrshort#1#2[#3] {%
3599 \glsdoifexists{#2}%
3600 {%
3601 \let\do@gls@link@checkfirsthyper\relax

3602 \def\glslabel{#2}%
3603 \let\glsifplural\@secondoftwo
3604 \let\glsapscase\@secondofthree
3605 \let\glsinsert\@empty
3606 \def\glscustomtext{%
3607 \acronymfont{\Glsentryshort{#2}}#3%
3608 }%
```

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

```
3609 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3610 }%
3611 }
```

\ACRshort

```
3612 \newrobustcmd*{\ACRshort}{\@gls@hyp@opt\@ns@ACRshort}
```

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

```
3613 \newcommand*{\ns@ACRshort}[2] [] {%
3614 \new@ifnextchar[{\@ACRshort{#1}{#2}}{\@ACRshort{#1}{#2} []}%
3615 }
```

Read in the final optional argument:

```
3616 \def\@ACRshort#1#2[#3] {%
3617 \glsdoifexists{#2}%
3618 {%
3619 \let\do@gls@link@checkfirsthyper\relax
```

```

3620 \def\glslabel{#2}%
3621 \let\glsifplural\@secondoftwo
3622 \let\glscapscase\@thirdofthree
3623 \let\glsinsert\@empty
3624 \def\glscustomtext{%
3625     \mfirstucMakeUppercase{\acronymfont{\glsentryshort{#2}}#3}%
3626 }%

```

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

```

3627 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3628 }%
3629 }

```

Short plural:

\acrshortpl

```

3630 \newrobustcmd*{\acrshortpl}{\@gls@hyp@opt\@ns@acrshortpl}

```

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

```

3631 \newcommand*{\ns@acrshortpl}[2] [] {%
3632     \new@ifnextchar[{\@acrshortpl{#1}{#2}}{\@acrshortpl{#1}{#2} []}%
3633 }

```

Read in the final optional argument:

```

3634 \def\@acrshortpl#1#2[#3] {%
3635     \glsdoifexists{#2}%
3636     {%
3637         \let\do@gls@link@checkfirsthyper\relax
3638
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         }%

```

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

```

3645 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3646 }%
3647 }

```

\Acrshortpl

```

3648 \newrobustcmd*{\Acrshortpl}{\@gls@hyp@opt\@ns@Acrshortpl}

```

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

```

3649 \newcommand*{\ns@Acrshortpl}[2] [] {%
3650     \new@ifnextchar[{\@Acrshortpl{#1}{#2}}{\@Acrshortpl{#1}{#2} []}%
3651 }

```

Read in the final optional argument:

```

3652 \def\@Acrshortpl#1#2[#3]{%
3653   \glsdoifexists{#2}%
3654   {%
3655     \let\do@gl@link@checkfirsthyper\relax

3656     \def\glslabel{#2}%
3657     \let\glsifplural\@firstoftwo
3658     \let\glscapscase\@secondofthree
3659     \let\glsinsert\@empty
3660     \def\glscustomtext{%
3661       \acronymfont{\Glsentryshortpl{#2}}#3%
3662     }%

```

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

```

3663   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3664 }%
3665 }

```

\ACRshortpl

```

3666 \newrobustcmd*{\ACRshortpl}{\@gl@hyp@opt\@ns@ACRshortpl}

```

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

```

3667 \newcommand*{\ns@ACRshortpl}[2][ ]{%
3668   \new@ifnextchar[{\@ACRshortpl{#1}{#2}}{\@ACRshortpl{#1}{#2}[]}%
3669 }

```

Read in the final optional argument:

```

3670 \def\@ACRshortpl#1#2[#3]{%
3671   \glsdoifexists{#2}%
3672   {%
3673     \let\do@gl@link@checkfirsthyper\relax

3674     \def\glslabel{#2}%
3675     \let\glsifplural\@firstoftwo
3676     \let\glscapscase\@thirdofthree
3677     \let\glsinsert\@empty
3678     \def\glscustomtext{%
3679       \mfirstucMakeUppercase{\acronymfont{\Glsentryshortpl{#2}}#3}%
3680     }%

```

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

```

3681   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3682 }%
3683 }

```

\acrlong

```

3684 \newrobustcmd*{\acrlong}{\@gl@hyp@opt\@ns@acrlong}

```



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

```
3685 \newcommand*{\ns@acrlong}[2] [] {%
3686   \new@ifnextchar[{\@acrlong{#1}{#2}}{\@acrlong{#1}{#2} []}%
3687 }
```

Read in the final optional argument:

```
3688 \def\@acrlong#1#2[#3] {%
3689   \glstoifexists{#2}%
3690   {%
3691     \let\do@gl@link@checkfirsthyper\relax

3692   \def\glslabel{#2}%
3693   \let\gl@ifplural\@secondoftwo
3694   \let\glscapscase\@firstofthree
3695   \let\glinsert\@empty
```

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

```
3696   \def\glscustomtext{%
3697     \glstrylong{#2}#3%
3698   }%
```

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

```
3699   \@gl@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
3700   }%
3701 }
```

\Acrlong

```
3702 \newrobustcmd*{\Acrlong}{\@gl@hyp@opt\ns@Acrlong}
```

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

```
3703 \newcommand*{\ns@Acrlong}[2] [] {%
3704   \new@ifnextchar[{\@Acrlong{#1}{#2}}{\@Acrlong{#1}{#2} []}%
3705 }
```

Read in the final optional argument:

```
3706 \def\@Acrlong#1#2[#3] {%
3707   \glstoifexists{#2}%
3708   {%
3709     \let\do@gl@link@checkfirsthyper\relax

3710   \def\glslabel{#2}%
3711   \let\gl@ifplural\@secondoftwo
3712   \let\glscapscase\@secondofthree
3713   \let\glinsert\@empty
```

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

```

3714 \def\glscustomtext{%
3715 \Glsentrylong{#2}#3%
3716 }%

```

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

```

3717 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3718 }%
3719 }

```

\ACRlong

```

3720 \newrobustcmd*{\ACRlong}{\@gls@hyp@opt\@ns@ACRlong}

```

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

```

3721 \newcommand*{\ns@ACRlong}[2][{}]{%
3722 \new@ifnextchar[{\@ACRlong{#1}{#2}}{\@ACRlong{#1}{#2}[]}%
3723 }

```

Read in the final optional argument:

```

3724 \def\@ACRlong#1#2[#3]{%
3725 \glsdoifexists{#2}%
3726 {%
3727 \let\do@gls@link@checkfirsthyper\relax

3728 \def\glslabel{#2}%
3729 \let\glsifplural\@secondoftwo
3730 \let\glsapscase\@thirdofthree
3731 \let\glsinsert\@empty

```

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

```

3732 \def\glscustomtext{%
3733 \mfirstucMakeUppercase{\Glsentrylong{#2}#3}%
3734 }%

```

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

```

3735 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3736 }%
3737 }

```

Short plural:

\acrlongpl

```

3738 \newrobustcmd*{\acrlongpl}{\@gls@hyp@opt\@ns@acrlongpl}

```

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

```

3739 \newcommand*{\ns@acrlongpl}[2][{}]{%
3740 \new@ifnextchar[{\@acrlongpl{#1}{#2}}{\@acrlongpl{#1}{#2}[]}%
3741 }

```

Read in the final optional argument:

```
3742 \def\@acrlongpl#1#2[#3]{%
3743   \glsdoifexists{#2}%
3744   {%
3745     \let\do@gl@link@checkfirsthyper\relax

3746     \def\glslabel{#2}%
3747     \let\glsifplural\@firstoftwo
3748     \let\glscapscase\@firstofthree
3749     \let\glsinsert\@empty
```

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

```
3750     \def\glscustomtext{%
3751       \glentrylongpl{#2}#3%
3752     }%
```

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

```
3753     \@gl@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
3754   }%
3755 }
```

\Acrlongpl

```
3756 \newrobustcmd*{\Acrlongpl}{\@gl@hyp@opt\ns@Acrlongpl}
```

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

```
3757 \newcommand*{\ns@Acrlongpl}[2][]{%
3758   \new@ifnextchar[{\@Acrlongpl{#1}{#2}}{\@Acrlongpl{#1}{#2}[]}%
3759 }
```

Read in the final optional argument:

```
3760 \def\@Acrlongpl#1#2[#3]{%
3761   \glsdoifexists{#2}%
3762   {%
3763     \let\do@gl@link@checkfirsthyper\relax

3764     \def\glslabel{#2}%
3765     \let\glsifplural\@firstoftwo
3766     \let\glscapscase\@secondofthree
3767     \let\glsinsert\@empty
```

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

```
3768     \def\glscustomtext{%
3769       \Glsentrylongpl{#2}#3%
3770     }%
```

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

```
3771     \@gl@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3772   }%
3773 }
```

`\ACRlongpl`

```
3774 \newrobustcmd*{\ACRlongpl}{\@gls@hyp@opt\ns@ACRlongpl}
```

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

```
3775 \newcommand*{\ns@ACRlongpl}[2] [] {%
3776   \new@ifnextchar[{\@ACRlongpl{#1}{#2}}{\@ACRlongpl{#1}{#2} []}%
3777 }
```

Read in the final optional argument:

```
3778 \def\@ACRlongpl#1#2[#3] {%
3779   \glsdoifexists{#2}%
3780   {%
3781     \let\do@gls@link@checkfirsthyper\relax
3782     \def\glslabel{#2}%
3783     \let\glsifplural\@firstoftwo
3784     \let\glscapscase\@thirdofthree
3785     \let\glsinsert\@empty
```

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

```
3786   \def\glscustomtext{%
3787     \mfirstucMakeUppercase{\glsentrylongpl{#2}{#3}%
3788   }%
```

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

```
3789   \@gls@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
3790   }%
3791 }
```

### 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}`

```
3792 \newcommand*{\@gls@entry@field}[2] {%
3793   \csname glo@glsdetoklabel{#1}@#2\endcsname
3794 }
```

`\glsletentryfield`

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

```

3795 \newcommand*\glsletentryfield}[3]{%
3796   \letcs{#1}{glo@\glsdetoklabel{#2}@#3}%
3797 }

```

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

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

```

3798 \newcommand*\@Gls@entry@field}[2]{%
3799   \letcs\@glo@text{glo@\glsdetoklabel{#1}@#2}%
3800   \xmakefirstuc{\@glo@text}%
3801 }

```

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

```

3802 \newcommand*\glsentryname}[1]{\@Gls@entry@field{#1}{name}}

```

\Glsentryname

```

3803 \newrobustcmd*\Glsentryname}[1]{%
3804   \@Gls@entry@field{#1}{name}%
3805 }

```

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

```

3806 \newcommand*\glsentrydesc}[1]{\@Gls@entry@field{#1}{desc}}

```

\Glsentrydesc

```

3807 \newrobustcmd*\Glsentrydesc}[1]{%
3808   \@Gls@entry@field{#1}{desc}%
3809 }

```

Plural form:

\glsentrydescplural

```

3810 \newcommand*\glsentrydescplural}[1]{%
3811   \@Gls@entry@field{#1}{descplural}%
3812 }

```

`\Glsentrydescplural`

```
3813 \newrobustcmd*{\Glsentrydescplural}[1]{%
3814   \@Gls@entry@field{#1}{descplural}%
3815 }
```

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`

```
3816 \newcommand*{\glsentrytext}[1]{\@Gls@entry@field{#1}{text}}
```

`\Glsentrytext`

```
3817 \newrobustcmd*{\Glsentrytext}[1]{%
3818   \@Gls@entry@field{#1}{text}%
3819 }
```

Get the plural form:

`\glsentryplural`

```
3820 \newcommand*{\glsentryplural}[1]{%
3821   \@Gls@entry@field{#1}{plural}%
3822 }
```

`\Glsentryplural`

```
3823 \newrobustcmd*{\Glsentryplural}[1]{%
3824   \@Gls@entry@field{#1}{plural}%
3825 }
```

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

`\glsentrysymbol`

```
3826 \newcommand*{\glsentrysymbol}[1]{%
3827   \@Gls@entry@field{#1}{symbol}%
3828 }
```

`\Glsentrysymbol`

```
3829 \newrobustcmd*{\Glsentrysymbol}[1]{%
3830   \@Gls@entry@field{#1}{symbol}%
3831 }
```

Plural form:

`\glsentrysymbolplural`

```
3832 \newcommand*{\glsentrysymbolplural}[1]{%
3833   \@Gls@entry@field{#1}{symbolplural}%
3834 }
```

lentrysymbolplural

```
3835 \newrobustcmd*{\Glsentrysymbolplural}[1]{%
3836   \@Gls@entry@field{#1}{symbolplural}%
3837 }
```

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).

\glentryfirst

```
3838 \newcommand*{\glentryfirst}[1]{%
3839   \@Gls@entry@field{#1}{first}%
3840 }
```

\Glsentryfirst

```
3841 \newrobustcmd*{\Glsentryfirst}[1]{%
3842   \@Gls@entry@field{#1}{first}%
3843 }
```

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

glentryfirstplural

```
3844 \newcommand*{\glentryfirstplural}[1]{%
3845   \@Gls@entry@field{#1}{firstpl}%
3846 }
```

Glsentryfirstplural

```
3847 \newrobustcmd*{\Glsentryfirstplural}[1]{%
3848   \@Gls@entry@field{#1}{firstpl}%
3849 }
```

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

\glentrytype

```
3850 \newcommand*{\glentrytype}[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.

\glentrysort

```
3851 \newcommand*{\glentrysort}[1]{%
3852   \@Gls@entry@field{#1}{sort}%
3853 }
```

\glentryuseri    Get the first user key (as specified by the user1 when the entry was defined).  
The argument is the label associated with the entry.

```
3854 \newcommand*{\glentryuseri}[1]{%
3855   \@Gls@entry@field{#1}{useri}%
3856 }
```

```

\Glsentryuseri
3857 \newrobustcmd*{\Glsentryuseri}[1]{%
3858   \@Gls@entry@field{#1}{useri}%
3859 }

\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.
3860 \newcommand*{\glsentryuserii}[1]{%
3861   \@Gls@entry@field{#1}{userii}%
3862 }

\Glsentryuserii
3863 \newrobustcmd*{\Glsentryuserii}[1]{%
3864   \@Gls@entry@field{#1}{userii}%
3865 }

\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.
3866 \newcommand*{\glsentryuseriii}[1]{%
3867   \@Gls@entry@field{#1}{useriii}%
3868 }

\Glsentryuseriii
3869 \newrobustcmd*{\Glsentryuseriii}[1]{%
3870   \@Gls@entry@field{#1}{useriii}%
3871 }

\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.
3872 \newcommand*{\glsentryuseriv}[1]{%
3873   \@Gls@entry@field{#1}{useriv}%
3874 }

\Glsentryuseriv
3875 \newrobustcmd*{\Glsentryuseriv}[1]{%
3876   \@Gls@entry@field{#1}{useriv}%
3877 }

\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.
3878 \newcommand*{\glsentryuserv}[1]{%
3879   \@Gls@entry@field{#1}{userv}%
3880 }

\Glsentryuserv
3881 \newrobustcmd*{\Glsentryuserv}[1]{%
3882   \@Gls@entry@field{#1}{userv}%
3883 }

```



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

```
3884 \newrobustcmd*{\glsentryuservi}[1]{%
3885   \@gls@entry@field{#1}{uservi}%
3886 }
```

`\Glsentryuservi`

```
3887 \newrobustcmd*{\Glsentryuservi}[1]{%
3888   \@Gls@entry@field{#1}{uservi}%
3889 }
```

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

```
3890 \newcommand*{\glsentryshort}[1]{\@gls@entry@field{#1}{short}}
```

`\Glsentryshort`

```
3891 \newrobustcmd*{\Glsentryshort}[1]{%
3892   \@Gls@entry@field{#1}{short}%
3893 }
```

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

```
3894 \newcommand*{\glsentryshortpl}[1]{\@gls@entry@field{#1}{shortpl}}
```

`\Glsentryshortpl`

```
3895 \newrobustcmd*{\Glsentryshortpl}[1]{%
3896   \@Gls@entry@field{#1}{shortpl}%
3897 }
```

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

```
3898 \newcommand*{\glsentrylong}[1]{\@gls@entry@field{#1}{long}}
```

`\Glsentrylong`

```
3899 \newrobustcmd*{\Glsentrylong}[1]{%
3900   \@Gls@entry@field{#1}{long}%
3901 }
```

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

```
3902 \newcommand*{\glsentrylongpl}[1]{\@gls@entry@field{#1}{longpl}}
```

`\Glsentrylongpl`

```
3903 \newrobustcmd*{\Glsentrylongpl}[1]{%
3904   \@Gls@entry@field{#1}{longpl}%
3905 }
```

Short cut macros to access full form:

`\glsentryfull`

```
3906 \newcommand*{\glsentryfull}[1]{%
3907   \acrfullformat{\glsentrylong{#1}}{\acronymfont{\glsentryshort{#1}}}%
3908 }
```

`\Glsentryfull`

```
3909 \newrobustcmd*{\Glsentryfull}[1]{%
3910   \acrfullformat{\Glsentrylong{#1}}{\acronymfont{\glsentryshort{#1}}}%
3911 }
```

`\glsentryfullpl`

```
3912 \newcommand*{\glsentryfullpl}[1]{%
3913   \acrfullformat{\glsentrylongpl{#1}}{\acronymfont{\glsentryshortpl{#1}}}%
3914 }
```

`\Glsentryfullpl`

```
3915 \newrobustcmd*{\Glsentryfullpl}[1]{%
3916   \acrfullformat{\Glsentrylongpl{#1}}{\acronymfont{\glsentryshortpl{#1}}}%
3917 }
```

`\glsentrynumberlist` Displays the number list as is.

```
3918 \newcommand*{\glsentrynumberlist}[1]{%
3919   \glsdoifexists{#1}%
3920   {%
3921     \@gls@entry@field{#1}{numberlist}%
3922   }%
3923 }
```

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

```
3924 \@ifpackageloaded{hyperref} {%
3925   \newcommand*{\glsdisplaynumberlist}[1]{%
3926     \GlossariesWarning
3927     {%
3928       \string\glsdisplaynumberlist\space
3929       doesn't work with hyperref.^^JUsing
3930       \string\glsentrynumberlist\space instead%
3931     }%
3932     \glsentrynumberlist{#1}%
3933   }%
3934 }%
3935 {%
3936   \newcommand*{\glsdisplaynumberlist}[1]{%
3937     \glsdoifexists{#1}%
3938     {%
3939       \bgroup
```



```
3975 \define@key{glossadd}{counter}{\def\@gls@counter{#1}}
```

```
3976 \define@key{glossadd}{format}{\def\@glsnumberformat{#1}}
```

This key is only used by `\glsaddall`:

```
3977 \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`

```
3978 \newrobustcmd*{\glsadd}[2][]{%
3979   \glsdoifexists{#2}%
3980   {%
3981     \def\@glsnumberformat{glsnumberformat}%
3982     \edef\@gls@counter{\csname glo@glsetoklabel{#2}@counter\endcsname}%
3983     \setkeys{glossadd}{#1}%
```

Store the entry's counter in `\theglentrycounter`

```
3984   \@gls@saveentrycounter
3985   \@do@wrglossary{#2}%
3986   }%
3987 }
```

**`\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`

```
3988 \newrobustcmd*{\glsaddall}[1][]{%
3989   \edef\@glo@type{\@glo@types}%
3990   \setkeys{glossadd}{#1}%
3991   \forallglsentries[\@glo@type]{\@glo@entry}{%
3992     \glsadd[#1]{\@glo@entry}%
3993   }%
3994 }
```

`\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.

```

3995 \newrobustcmd*{\glsaddallunused}[1][\@glo@types]{%
3996 \forallglsentries[#1]{\@glo@entry}%
3997 {%
3998 \ifglsused{\@glo@entry}{\glsadd[format=@gobble]{\@glo@entry}}%
3999 }%
4000 }

```

## 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 `\glssetgrouptitle` 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.

```
4001 \edef\glsopenbrace{\expandafter\@gobble\string\{}
```

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

```
4002 \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.

```
4003 \edef\glsquote#1{\string"#1\string"}
```

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

```

4004 \ifglsxindy
4005 \newcommand*{\@glsfirstletter}{A}
4006 \fi

```

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

```

4007 \ifglsxindy
4008 \newcommand*{\GlsSetXdyFirstLetterAfterDigits}[1]{%
4009 \renewcommand*{\@glsfirstletter}{#1}}
4010 \else

```

```

4011 \newcommand*{\GlsSetXdyFirstLetterAfterDigits}[1]{%
4012 \glsnoxindywarning\GlsSetXdyFirstLetterAfterDigits}
4013 \fi

\@glsminrange Define the minimum number of successive location references to merge into a
range.
4014 \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.
4015 \ifglxindy
4016 \newcommand*{\GlsSetXdyMinRangeLength}[1]{%
4017 \renewcommand*{\@glsminrange}{#1}}
4018 \else
4019 \newcommand*{\GlsSetXdyMinRangeLength}[1]{%
4020 \glsnoxindywarning\GlsSetXdyMinRangeLength}
4021 \fi

\writeist
4022 \ifglxindy
Code to use if xindy is required.
4023 \def\writeist{%
Define write register if not already defined
4024 \ifundef{\glswrite}{\newwrite\glswrite}{}%
Update attributes list
4025 \@gls@addpredefinedattributes
Open the file.
4026 \openout\glswrite=\istfilename
Write header comment at the start of the file
4027 \write\glswrite{;; xindy style file created by the glossaries
4028 package}%
4029 \write\glswrite{;; for document '\jobname' on
4030 \the\year-\the\month-\the\day}%
Specify the required styles
4031 \write\glswrite{^^J; required styles^^J}
4032 \@for\@xdystyle:=\@xdyrequiredstyles\do{%
4033 \ifx\@xdystyle\@empty
4034 \else
4035 \protected@write\glswrite{ }(require
4036 \string"\@xdystyle.xdy\string")}%
4037 \fi
4038 }%

```

List the allowed attributes (possible values used by the format key)

```
4039 \write\glswrite{^^J%
4040 ; list of allowed attributes (number formats)^^J}%
4041 \write\glswrite{(define-attributes ((\@xdyattributes)))}%
```

Define any additional alphabets

```
4042 \write\glswrite{^^J; user defined alphabets^^J}%
4043 \write\glswrite{\@xdyuseralphabets}%
```

Define location classes.

```
4044 \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.

```
4045 \@for\@gls@classI:=\@gls@xdy@locationlist\do{%
```

Case were  $\langle Hprefix \rangle$  is empty:

```
4046 \protected@write\glswrite{}{(define-location-class
4047 \string"\@gls@classI\string"^^J\space\space\space
4048 (
4049 :sep "{{"
4050 \csname @gls@xdy@Lclass@\@gls@classI\endcsname\space
4051 :sep "}"
4052 )
4053 ^^J\space\space\space
4054 :min-range-length \@glsminrange^^J%
4055 )
4056 }%
```

Nested iteration over all classes:

```
4057 {%
4058 \@for\@gls@classII:=\@gls@xdy@locationlist\do{%
4059 \protected@write\glswrite{}{(define-location-class
4060 \string"\@gls@classII-\@gls@classI\string"
4061 ^^J\space\space\space
4062 (
4063 :sep "{"
4064 \csname @gls@xdy@Lclass@\@gls@classII\endcsname\space
4065 :sep "{{"
4066 \csname @gls@xdy@Lclass@\@gls@classI\endcsname\space
4067 :sep "}"
4068 )
4069 ^^J\space\space\space
4070 :min-range-length \@glsminrange^^J%
4071 )
4072 }%
4073 }%
4074 }%
4075 }%
```

User defined location classes (needs checking for new location format).

```

4076 \write\glswrite{^^J; user defined location classes}%
4077 \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.)

```

4078 \write\glswrite{^^J; define cross-reference class^^J}%
4079 \write\glswrite{(define-crossref-class \string"see\string"
4080 :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.)

```

4081 \write\glswrite{(markup-crossref-list
4082 :class \string"see\string"^^J\space\space\space
4083 :open \string"\string\glsseeformat\string"
4084 :close \string"{}\string")}%

```

List the order to sort the classes.

```

4085 \write\glswrite{^^J; define the order of the location classes}%
4086 \write\glswrite{(define-location-class-order
4087 (\@xdylocationclassorder))}%

```

Specify what to write to the start and end of the glossary file.

```

4088 \write\glswrite{^^J; define the glossary markup^^J}%

4089 \write\glswrite{(markup-index^^J\space\space\space
4090 :open \string"\string
4091 \glossarysection[\string\glossarytoctitle]{\string
4092 \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`)

```

4093 \@for\@this@ctr:=\@xdycounters\do{%
4094 {%
4095   \@for\@this@attr:=\@xdyattributelist\do{%
4096     \protected\write\glswrite{{}\string\providecommand*%
4097       \expandafter\string
4098       \csname glsX\@this@ctr X\@this@attr\endcsname[2]%
4099       {%
4100         \string\setentrycounter
4101         [\expandafter\@gobble\string\#1]{\@this@ctr}%
4102         \expandafter\string
4103         \csname\@this@attr\endcsname
4104         {\expandafter\@gobble\string\#2}%
4105       }%
4106     }%
4107   }%
4108 }%
4109 }%

```



Add the end part of the open tag and the rest of the markup-index information:

```
4110 \write\glswrite{%
4111   \string\begin
4112   {theglossary}\string\glossaryheader\string~n\string" ^^J\space
4113   \space\space:close \string"\expandafter\@gobble
4114   \string~n\string~n\string
4115   \end{theglossary}\string\glossarypostamble
4116   \string~n\string" ^^J\space\space\space
4117   :tree)}}%
```

Specify what to put between letter groups

```
4118 \write\glswrite{(markup-letter-group-list
4119   :sep \string"\string\glsgroupskip\string~n\string"))}%
```

Specify what to put between entries

```
4120 \write\glswrite{(markup-indexentry
4121   :open \string"\string\relax \string\glsresetentrylist
4122   \string~n\string"))}%
```

Specify how to format entries

```
4123 \write\glswrite{(markup-locclass-list :open
4124   \string"\glsopenbrace\string\glossaryentrynumbers
4125   \glsopenbrace\string\relax\space \string"^^J\space\space\space
4126   :sep \string", \string"
4127   :close \string"\glsclosebrace\glsclosebrace\string"))}%
```

Specify how to separate location numbers

```
4128 \write\glswrite{(markup-locref-list
4129   :sep \string"\string\delimN\space\string"))}%
```

Specify how to indicate location ranges

```
4130 \write\glswrite{(markup-range
4131   :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.

```
4132 \@onelevel@sanitize\gls@suffixF
4133 \@onelevel@sanitize\gls@suffixFF
4134 \ifx\gls@suffixF\@empty
4135 \else
4136   \write\glswrite{(markup-range
4137     :close "\gls@suffixF" :length 1 :ignore-end)}}%
4138 \fi
4139 \ifx\gls@suffixFF\@empty
4140 \else
4141   \write\glswrite{(markup-range
4142     :close "\gls@suffixFF" :length 2 :ignore-end)}}%
4143 \fi
```

Specify how to format locations.

```
4144 \write\glswrite{^^J; define format to use for locations^^J}%
4145 \write\glswrite{\@xdylocref}}%
```

Specify how to separate letter groups.

```
4146 \write\glswrite{^^J; define letter group list format^^J}%
4147 \write\glswrite{(markup-letter-group-list
4148 :sep \string"\string\glsgroupskip\string~n\string")}%
```

Define letter group headings.

```
4149 \write\glswrite{^^J; letter group headings^^J}%
4150 \write\glswrite{(markup-letter-group
4151 :open-head \string"\string\glsgroupheading
4152 \glsoopenbrace\string"^^J\space\space\space
4153 :close-head \string"\glsclosebrace\string")}%
```

Define additional letter groups.

```
4154 \write\glswrite{^^J; additional letter groups^^J}%
4155 \write\glswrite{\@xdylettergroups}%
```

Define additional sort rules

```
4156 \write\glswrite{^^J; additional sort rules^^J}
4157 \write\glswrite{\@xdysortrules}%
```

Close the style file

```
4158 \closeout\glswrite
```

Suppress any further calls.

```
4159 \let\writeist\relax
4160 }
4161 \else
```

Code to use if makeindex is required.

```
4162 \edef\@gls@actualchar{\string?}
4163 \edef\@gls@encapchar{\string|}
4164 \edef\@gls@levelchar{\string!}
4165 \edef\@gls@quotechar{\string"}
4166 \def\writeist{\relax
4167 \ifundef\glswrite{\newwrite\glswrite}{\relax
4168 \openout\glswrite=\istfilename
4169 \write\glswrite{\expandafter\@gobble\string}% makeindex style file
4170 created by the glossaries package}
4171 \write\glswrite{\expandafter\@gobble\string}% for document
4172 '\jobname' on \the\year-\the\month-\the\day}
4173 \write\glswrite{actual '\@gls@actualchar'}
4174 \write\glswrite{encap '\@gls@encapchar'}
4175 \write\glswrite{level '\@gls@levelchar'}
4176 \write\glswrite{quote '\@gls@quotechar'}
4177 \write\glswrite{keyword \string"\string\glossaryentry\string"}
4178 \write\glswrite{preamble \string"\string\glossarysection[\string
4179 \glossarytoctitle]{\string\glossarytitle}\string
4180 \glossarypreamble\string\n\string\begin{theglossary}\string
4181 \glossaryheader\string\n\string"}
4182 \write\glswrite{postamble \string"\string%\string\n\string
4183 \end{theglossary}\string\glossarypostamble\string\n
4184 \string"}
```

```

4185 \write\glswrite{group_skip \string"\string\glsgroupskip\string\n
4186 \string"}
4187 \write\glswrite{item_0 \string"\string%\string\n\string"}
4188 \write\glswrite{item_1 \string"\string%\string\n\string"}
4189 \write\glswrite{item_2 \string"\string%\string\n\string"}
4190 \write\glswrite{item_01 \string"\string%\string\n\string"}
4191 \write\glswrite{item_x1
4192 \string"\string\relax \string\glresetentrylist\string\n
4193 \string"}
4194 \write\glswrite{item_12 \string"\string%\string\n\string"}
4195 \write\glswrite{item_x2
4196 \string"\string\relax \string\glresetentrylist\string\n
4197 \string"}

4198 \write\glswrite{delim_0 \string"\string\{\string
4199 \glssymbols\string\{\string\relax \string"}
4200 \write\glswrite{delim_1 \string"\string\{\string
4201 \glssymbols\string\{\string\relax \string"}
4202 \write\glswrite{delim_2 \string"\string\{\string
4203 \glssymbols\string\{\string\relax \string"}
4204 \write\glswrite{delim_t \string"\string\}\string\}\string"}
4205 \write\glswrite{delim_n \string"\string\delimN \string"}
4206 \write\glswrite{delim_r \string"\string\delimR \string"}
4207 \write\glswrite{headings_flag 1}
4208 \write\glswrite{heading_prefix
4209 \string"\string\glsgroupheading\string\{\string"}
4210 \write\glswrite{heading_suffix
4211 \string"\string\}\string\relax
4212 \string\glresetentrylist \string"}
4213 \write\glswrite{symhead_positive \string"glssymbols\string"}
4214 \write\glswrite{numhead_positive \string"glssymbols\string"}
4215 \write\glswrite{page_compositor \string"glscpositor\string"}
4216 \@glscbsdq\glscbsdq\glscbsdq\glscbsdq\glscbsdq\glscbsdq
4217 \@glscbsdq\glscbsdq\glscbsdq\glscbsdq\glscbsdq\glscbsdq
4218 \ifx\glscbsdq\glscbsdq\glscbsdq\glscbsdq\glscbsdq\glscbsdq
4219 \else
4220 \write\glswrite{suffix_2p \string"\glscbsdq\glscbsdq\glscbsdq\glscbsdq\glscbsdq\glscbsdq"}
4221 \fi
4222 \ifx\glscbsdq\glscbsdq\glscbsdq\glscbsdq\glscbsdq\glscbsdq
4223 \else
4224 \write\glswrite{suffix_3p \string"\glscbsdq\glscbsdq\glscbsdq\glscbsdq\glscbsdq\glscbsdq"}
4225 \fi
4226 \closeout\glswrite
4227 \let\writeist\relax
4228 }
4229 \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`

```
4230 \newcommand{\noist}{%
```

Update attributes list

```
4231 \@gls@addpredefinedattributes
```

```
4232 \let\writeist\relax
```

```
4233 }
```

`\@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`

```
4234 \newcommand*{\@makeglossary}[1]{%
```

```
4235 \ifglossaryexists{#1}%
```

```
4236 {%
```

Only create a new write if `savewrites=false` otherwise create a token to collect the information.

```
4237 \ifglssavewrites
```

```
4238 \expandafter\newtoks\csname glo@#1@filetok\endcsname
```

```
4239 \else
```

```
4240 \expandafter\newwrite\csname glo@#1@file\endcsname
```

```
4241 \expandafter\@glsopenfile\csname glo@#1@file\endcsname{#1}%
```

```
4242 \fi
```

```
4243 \@gls@renewglossary
```

```
4244 \writeist
```

```
4245 }%
```

```
4246 {%
```

```
4247 \PackageError{glossaries}%
```

```
4248 {Glossary type ‘#1’ not defined}%
```

```
4249 {New glossaries must be defined before using \string\makeglossary}%
```

```
4250 }%
```

```
4251 }
```

`\@glsopenfile` Open write file associated with the given glossary.

```
4252 \newcommand*{\@glsopenfile}[2]{%
```

```
4253 \immediate\openout#1=\jobname.\csname @glotype@#2@out\endcsname
```

```
4254 \PackageInfo{glossaries}{Writing glossary file
```

```
4255 \jobname.\csname @glotype@#2@out\endcsname}%
```

```
4256 }
```

\@closegls

```
4257 \newcommand*{\@closegls}[1]{%
4258   \closeout\csname glo@#1@file\endcsname
4259 }
4260 %   \end{macrocode}
4261 %\end{macro}
4262 %
4263 %\begin{macro}{\@gls@automake}
4264 %\changes{4.08}{2014-07-30}{new}
4265 %   \begin{macrocode}
4266 \ifglxindy
4267   \newcommand*{\@gls@automake}[1]{%
4268     \ifglossaryexists{#1}
4269     {%
4270       \@closegls{#1}%
4271       \ifdefstring{glsorder}{letter}%
4272         {\def\@gls@order{-M ord/letorder }}%
4273         {\let\@gls@order\@empty}%
4274       \ifcsundef{xdy@#1@language}%
4275         {\let\@gls@langmod\@xdy@main@language}%
4276         {\letcs\@gls@langmod{xdy@#1@language}}%
4277       \edef\@gls@dothiswrite{\noexpand\write18{xindy
4278         -I xindy
4279         \@gls@order
4280         -L \@gls@langmod\space
4281         -M \@gls@istfilebase\space
4282         -C \@gls@codepage\space
4283         -t \jobname.\csuse{@glotype@#1@log}
4284         -o \jobname.\csuse{@glotype@#1@in}
4285         \jobname.\csuse{@glotype@#1@out}}}%
4286     }%
4287     \@gls@dothiswrite
4288   }%
4289   {%
4290     \GlossariesWarning{Can't make glossary '#1', it doesn't exist}%
4291   }%
4292 }
4293 \else
4294   \newcommand*{\@gls@automake}[1]{%
4295     \ifglossaryexists{#1}
4296     {%
4297       \@closegls{#1}%
4298       \ifdefstring{glsorder}{letter}%
4299         {\def\@gls@order{-l }}%
4300         {\let\@gls@order\@empty}%
4301       \edef\@gls@dothiswrite{\noexpand\write18{makeindex \@gls@order
4302         -s \istfilename\space
4303         -t \jobname.\csuse{@glotype@#1@log}
4304         -o \jobname.\csuse{@glotype@#1@in}}
```

```

4305     \jobname.\csuse{@glotype@#1@out}}%
4306   }%
4307   \@gls@dothiswrite
4308 }%
4309 {%
4310   \GlossariesWarning{Can't make glossary '#1', it doesn't exist}%
4311 }%
4312 }
4313 \fi

```

`\nomakeglossaries` Issue warning that `\makeglossaries` hasn't been used.

```

4314 \newcommand*{\@warn@nomakeglossaries}{%

```

Only use this if warning if `\printglossary` has been used without `\makeglossaries`

```

4315 \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`

```

4316 \newcommand*{\makeglossaries}{%

```

Define the write used for style file also used for all other output files if `savewrites=true`.

```

4317 \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.

```

4318 \protected@write\@auxout{}{\string\providecommand\string\@glsorder[1]{}%
4319 \protected@write\@auxout{}{\string\providecommand\string\@istfilename[1]{}%

```

Write the name of the style file to the aux file (needed by `makeglossaries`)

```

4320 \protected@write\@auxout{}{\string\@istfilename{\istfilename}}%
4321 \protected@write\@auxout{}{\string\@glsorder{\glsorder}}

```

Iterate through each glossary type and activate it.

```

4322 \@for\@glo@type:=\@glo@types\do{%
4323   \ifthenelse{\equal{\@glo@type}{}}{}{}%
4324   \@makeglossary{\@glo@type}}%
4325 }%

```

New glossaries must be created before `\makeglossaries` so disable `\newglossary`.

```

4326 \renewcommand*\newglossary[4][]{%
4327 \PackageError{glossaries}{New glossaries
4328 must be created before \string\makeglossaries}{You need
4329 to move \string\makeglossaries\space after all your
4330 \string\newglossary\space commands}}%

```

Any subsequence instances of this command should have no effect

```
4331 \let\@makeglossary\relax
4332 \let\makeglossary\relax
4333 \let\makeglossaries\relax
```

Disable all commands that have no effect after \makeglossaries

```
4334 \@disable@onlypremakeg
```

Allow see key:

```
4335 \let\gls@checkseeallowed\relax
```

Suppress warning about no \makeglossaries

```
4336 \let\warn@nomakeglossaries\relax
```

Activate warning about missing \printglossary

```
4337 \def\warn@noprntglossary{%
4338   \GlossariesWarningNoLine{No \string\printglossary\space
4339     or \string\printglossaries\space
4340     found.^^J(Remove \string\makeglossaries\space if you don't want
4341     any glossaries.)^^JThis document will not have a glossary}%
4342 }%
```

Declare list parser for \glsdisplaynumberlist

```
4343 \ifglssavenumberlist
4344   \edef\@gls@dodolistparser{\noexpand\DeclareListParser
4345     {\noexpand\glsnumlistparser}{\delimN}}}%
4346   \@gls@dodolistparser
4347 \fi
```

Prevent user from also using \makenoidxglossaries

```
4348 \let\makenoidxglossaries\@no@makeglossaries
```

Prohibit sort key in printgloss family:

```
4349 \renewcommand*{\@printgloss@setsort}{}%
4350 \let\@glo@assign@sortkey\@glo@no@assign@sortkey
4351 }%
```

Check the automake setting:

```
4352 \ifglssautomake
4353   \renewcommand*{\@gls@doautomake}{%
4354     \@for\@gls@type:=\@glo@types\do{%
4355       \ifdefempty{\@gls@type}{}%
4356       {\@gls@automake{\@gls@type}}%
4357     }%
4358   }%
4359 \fi
4360 }
```

Must occur in the preamble:

```
4361 \@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`

```
4362 \let\makeglossary\makeglossaries
```

If `\makeglossaries` hasn't been used, issue a warning. Also issue a warning  
if neither `\printglossaries` nor `\printglossary` have been used.

```
4363 \AtEndDocument{%
4364   \warn@nomakeglossaries
4365   \warn@noprintglossary
4366 }
```

`\makenoidxglossaries` Analogous to `\makeglossaries` this activates the commands needed for `\printnoidxglossary`

```
4367 \newcommand*\makenoidxglossaries{%
```

Redefine empty glossary warning:

```
4368   \renewcommand{\@gls@noref@warn}[1]{%
4369     \GlossariesWarning{Empty glossary for
4370     \string\printnoidxglossary[type={##1}].
4371     Rerun may be required (or you may have forgotten to use
4372     commands like \string\gls).}%
4373   }%
```

Don't escape makeindex/xindy characters

```
4374   \let\@gls@checkmkidxchars\@gobble
```

Write glossary information to aux instead of glossary files

```
4375   \let\@@do@@wrglossary\gls@noidxglossary
```

Switch on group headings that use the character code:

```
4376   \let\@gls@getgrouptitle\@gls@noidx@getgrouptitle
```

Allow see key:

```
4377   \let\gls@checkseeallowed\relax
```

Redefine cross-referencing macro:

```
4378   \renewcommand{\@do@seeglossary}[2]{%
4379     \edef\@gls@label{\glsdetoklabel{##1}}%
4380     \protected@write\@auxout{%
4381       \string\@gls@reference
4382       {\csname glo@\@gls@label @type\endcsname}%
4383       {\@gls@label}%
4384       {%
4385         \string\glsseeformat##2}%
4386       }%
4387     }%
4388   }%
```



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.

```

4389 \AtBeginDocument
4390 {%
4391 \write\@auxout{\string\providecommand\string\@gls@reference[3]{}}%
4392 }%

Change warning about no glossares
4393 \def\warn@noprintglossary{%
4394 \GlossariesWarningNoLine{No \string\printnoidxglossary\space
4395 or \string\printnoidxglossaries ^^J
4396 found. (Remove \string\makenoidxglossaries\space if you
4397 don't want any glossaries.)^^JThis document will not have a glossary}%
4398 }%

Suppress warning about no \makeglossaries
4399 \let\warn@nomakeglossaries\relax

Prevent user from also using \makeglossaries
4400 \let\makeglossaries\@no@makeglossaries

Allow sort key in printgloss family:
4401 \renewcommand*{\@printgloss@setsort}{%
4402 \let\@glo@assign@sortkey\@glo@assign@sortkey

Initialise default sort order:
4403 \def\@glo@sorttype{\@glo@default@sorttype}%
4404 }%

All entries must be defined in the preamble:
4405 \renewcommand*\new@glossaryentry[2]{%
4406 \PackageError{glossaries}{Glossary entries must be
4407 defined in the preamble^^Jwhen you use
4408 \string\makenoidxglossaries}%
4409 {Either move your definitions to the preamble or use
4410 \string\makeglossaries}%
4411 }%

Redefine \glsentrynumberlist
4412 \renewcommand*{\glsentrynumberlist}[1]{%
4413 \letcs{\@gls@loclist}{\glo@\glsdetoklabel{##1}@loclist}%
4414 \ifdef\@gls@loclist
4415 {%
4416 \glsnoidxloclist{\@gls@loclist}%
4417 }%
4418 {%
4419 \ifglsentryexists{##1}%
4420 {%
4421 \GlossariesWarning{Missing location list for ‘##1’. Either
4422 a rerun is required or you haven't referenced the entry.}%
4423 }%

```

```

4424     {%
4425         \PackageError{glossaries}{Glossary entry ‘##1’ has not been
4426             defined.}{}%
4427     }%
4428 }%
4429 }%

Redefine \glsdisplaynumberlist
4430 \renewcommand*{\glsdisplaynumberlist}[1]{%
4431     \letcs{\@gls@loclist}{glo@\glsdetoklabel{##1}@loclist}%
4432     \ifdef\@gls@loclist
4433     {%
4434         \def\@gls@noidxloclist@sep{%
4435             \def\@gls@noidxloclist@sep{%
4436                 \def\@gls@noidxloclist@sep{%
4437                     \glsnumlistsep
4438                 }%
4439                 \def\@gls@noidxloclist@finalsep{\glsnumlistlastsep}%
4440             }%
4441         }%
4442         \def\@gls@noidxloclist@finalsep{}%
4443         \def\@gls@noidxloclist@prev{}%
4444         \forlistloop{\glsnoidxdisplayloclisthandler}{\@gls@loclist}%
4445         \@gls@noidxloclist@finalsep
4446         \@gls@noidxloclist@prev
4447     }%
4448     {%
4449         ??\ifglsentryexists{##1}%
4450         {%
4451             \GlossariesWarning{Missing location list for ‘##1’. Either
4452                 a rerun is required or you haven’t referenced the entry.}%
4453         }%
4454         {%
4455             \PackageError{glossaries}{Glossary entry ‘##1’ has not been
4456                 defined.}{}%
4457         }%
4458     }%
4459 }%

```

Provide a generic way of iterating through the number list:

```

4460 \renewcommand*{\glsnumberlistloop}[3]{%
4461     \letcs{\@gls@loclist}{glo@\glsdetoklabel{##1}@loclist}%
4462     \let\@gls@org@glsnoidxdisplayloc\glsnoidxdisplayloc
4463     \let\@gls@org@glsseeformat\glsseeformat
4464     \let\glsnoidxdisplayloc##2\relax
4465     \let\glsseeformat##3\relax
4466     \ifdef\@gls@loclist
4467     {%
4468         \forlistloop{\glsnoidxnumberlistloophandler}{\@gls@loclist}%
4469     }%

```

```

4470   {%
4471     \ifglsentryexists{##1}%
4472     {%
4473       \GlossariesWarning{Missing location list for ‘##1’. Either
4474       a rerun is required or you haven’t referenced the entry.}%
4475     }%
4476     {%
4477       \PackageError{glossaries}{Glossary entry ‘##1’ has not been
4478       defined.}{}%
4479     }%
4480   }%
4481   \let\glsnoidxdisplayloc\@gls@org\glsnoidxdisplayloc
4482   \let\glsseeformat\@gls@org\glsseeformat
4483 }%

```

Modify sanitize sort function

```

4484 \let\@gls@sanitizesort\@gls@noidx@sanitizesort
4485 \let\@gls@nosanitizesort\@gls@noidx@nosanitizesort
4486 \@gls@noidx@setsanitizesort
4487 }

```

Preamble-only command:

```

4488 \onlypreamble{\makenoidxglossaries}

```

`\glsnumberlistloop` `\glsnumberlistloop{<label>}{<handler>}`

```

4489 \newcommand*{\glsnumberlistloop}[2]{%
4490   \PackageError{glossaries}{\string\glsnumberlistloop\space
4491   only works with \string\makenoidxglossaries}{}%
4492 }

```

`numberlistloophandler` Handler macro for `\glsnumberlistloop`. (The argument should be in the form `\glsnoidxdisplayloc{<prefix>}{<counter>}{<format>}{<n>}`)

```

4493 \newcommand*{\glsnoidxnumberlistloophandler}[1]{%
4494   #1%
4495 }

```

`\@no@makeglossaries` Can’t use both `\makeglossaries` and `\makenoidxglossaries`

```

4496 \newcommand*{\@no@makeglossaries}{%
4497   \PackageError{glossaries}{You can’t use both
4498   \string\makeglossaries\space and \string\makenoidxglossaries}%
4499   {Either use one or other (or none) of those commands but not both
4500   together.}%
4501 }

```

`\@gls@noref@warn` Warning when no instances of `\@gls@reference` found.

```

4502 \newcommand{\@gls@noref@warn}[1]{%
4503   \GlossariesWarning{\string\makenoidxglossaries\space

```

```

4504   is required to make \string\printnoidxglossary[type={#1}] work}%
4505 }

```

`\gls@noidxglossary` Write the glossary information to the aux file:

```

4506 \newcommand*{\gls@noidxglossary}{%
4507   \protected@write\@auxout{}{%
4508     \string\@gls@reference
4509     {\csname glo@\@gls@label @type\endcsname}%
4510     {\@gls@label}%
4511     {\string\glsnoidxdisplayloc
4512      {\@glo@counterprefix}%
4513      {\@gls@counter}%
4514      {\@glsnumberformat}%
4515      {\@glslocref}%
4516     }%
4517   }%
4518 }

```

### 1.13 Writing information to associated files

`\istfile` Deprecated.

```

4519 \def\istfile{\glswrite}

```

At the end of the document, the files should be created if `savewrites=true`.

```

4520 \AtEndDocument{%
4521   \glswritefiles
4522 }

```

`\@glswritefiles` Only write the files if `savewrites=true`

```

4523 \newcommand*{\@glswritefiles}{%

```

Iterate through all the glossaries

```

4524   \foralllglossaries{\@glo@type}{%

```

Check for empty glossaries (patch provided by Patrick Häcker)

```

4525     \ifcsundef{glo@\@glo@type @filetok}%
4526     {%
4527       \def\gls@tmp{}%
4528     }%
4529     {%
4530       \edef\gls@tmp{\expandafter\the
4531         \csname glo@\@glo@type @filetok\endcsname}%
4532     }%
4533     \ifx\gls@tmp\@empty
4534       \ifx\@glo@type\glsdefaulttype
4535         \GlossariesWarningNoLine{Glossary '\@glo@type' has no
4536           entries.^^JRemember to use package option 'nomain' if
4537 you
4538           don't want to^^Juse the main glossary}%

```

```

4539         \else
4540             \GlossariesWarningNoLine{Glossary ‘\@glo@type’ has no
4541                 entries}%
4542         \fi
4543     \else
4544         \@glsopenfile{\glswrite}{\@glo@type}%
4545         \immediate\write\glswrite{%
4546             \expandafter\the
4547             \csname glo@\@glo@type @filetok\endcsname}%
4548         \immediate\closeout\glswrite
4549     \fi
4550 }%
4551 }

```

The `\glossary` command is redefined so that it takes an optional argument *<type>* to specify the glossary type (use `\glsdefaulttype` glossary by default). This shouldn't be used at user level as `\glslink` sets the correct format. The associated number should be stored in `\theglsentrycounter` before using `\glossary`.

`\glossary`

```

4552 \renewcommand*{\glossary}[1][\glsdefaulttype]{%
4553     \@glossary[#1]%
4554 }

```

Define internal `\@glossary` to ignore its argument. This gets redefined in `\@makeglossary`. This is defined to just `\index` as memoir changes the definition of `\@index`. (Thanks to Dan Luecking for pointing this out.)

`\@glossary`

```

4555 \def\@glossary[#1]{\index}

```

This is a convenience command to set `\@glossary`. It is used by `\@makeglossary` and then redefined to do nothing, as it only needs to be done once.

`\@gls@renewglossary`

```

4556 \newcommand{\@gls@renewglossary}{%
4557     \gdef\@glossary[##1]{\@bsphack\begingroup\@wrglossary{##1}}%
4558     \let\@gls@renewglossary\@empty
4559 }

```

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`

```

4560 \renewcommand*{\@wrglossary}[2]{%
4561     \ifglssavewrites
4562         \protected@edef\@gls@tmp{\the\csname glo@#1@filetok\endcsname#2}%

```

```

4563 \expandafter\global\expandafter\csname glo@#1@filetok\endcsname
4564 \expandafter{\@gls@tmp^^J}%
4565 \else

4566 \ifcsdef{glo@#1@file}%
4567 {%
4568 \expandafter\protected@write\csname glo@#1@file\endcsname{%
4569 \gls@disablepagerefexpansion}{#2}%
4570 }%
4571 {%
4572 \ifignoredglossary{#1}{}%
4573 {%
4574 \GlossariesWarning{No file defined for glossary ‘#1’}%
4575 }%
4576 }%
4577 \fi
4578 \endgroup\@esphack
4579 }

```

\@do@wrglossary

```

4580 \newcommand*{\@do@wrglossary}[1]{%
4581 \ifglsindexonlyfirst
4582 \ifglsused{#1}{\@do@wrglossary{#1}}%
4583 \else
4584 \@do@wrglossary{#1}%
4585 \fi
4586 }

```

@protected@pagefmts List of page formats to be protected against expansion.

```

4587 \newcommand{\gls@protected@pagefmts}{%
4588 \gls@numberpage,\gls@alphpage,\gls@Alphpage,\gls@romanpage,\gls@Romanpage%
4589 }

```

blepagerefexpansion

```

4590 \newcommand*{\gls@disablepagerefexpansion}{%
4591 \@for\@gls@this:=\gls@protected@pagefmts\do
4592 {%
4593 \expandafter\let\@gls@this\relax
4594 }%
4595 }

```

\gls@alphpage

```

4596 \newcommand*{\gls@alphpage}{\@alph\c@page}

```

\gls@Alphpage

```

4597 \newcommand*{\gls@Alphpage}{\@Alph\c@page}

```

\gls@numberpage

```

4598 \newcommand*{\gls@numberpage}{\number\c@page}

```

```

\gls@romanpage
4599 \newcommand*{\gls@romanpage}{\romannumeral\c@page}

\gls@Romanpage
4600 \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.
4601 \newcommand*{\@do@wrglossary}[1]{%
4602   \begingroup

   First a bit of hackery to prevent premature expansion of \c@page. Store original
   definitions:
4603   \let\orgthe\the
4604   \let\orgnumber\number
4605   \let\orgromannumeral\romannumeral
4606   \let\orgalph\@alph
4607   \let\orgAlph\@Alph
4608   \let\orgRoman\@Roman

   Redefine:
4609   \def\the##1{%
4610     \ifx##1\c@page \gls@numberpage\else\orgthe##1\fi}%
4611   \def\number##1{%
4612     \ifx##1\c@page \gls@numberpage\else\orgnumber##1\fi}%
4613   \def\romannumeral##1{%
4614     \ifx##1\c@page \gls@romanpage\else\orgromannumeral##1\fi}%
4615   \def\@Roman##1{%
4616     \ifx##1\c@page \gls@Romanpage\else\orgRoman##1\fi}%
4617   \def\@alph##1{%
4618     \ifx##1\c@page \gls@alphpage\else\orgalph##1\fi}%
4619   \def\@Alph##1{%
4620     \ifx##1\c@page \gls@Alphpage\else\orgAlph##1\fi}%

   Prevent expansion:
4621   \gls@disablepagerefexpansion

   Now store location in \glslocref:
4622   \protected@xdef\glslocref{\theHglentrycounter}%
4623   \endgroup

   Escape any special characters
4624   \@gls@checkmkidxchars\glslocref

   Check if the hyper-location is the same as the location and set the hyper prefix.

4625   \expandafter\ifx\theHglentrycounter\theHglentrycounter\relax
4626   \def\@glo@counterprefix{%
4627   \else
4628     \protected@edef\@glsHlocref{\theHglentrycounter}%
4629     \@gls@checkmkidxchars\@glsHlocref

```

```

4630 \edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix
4631 {\@gls@locref}{\@gls@Hlocref}}%
4632 }%
4633 \@do@gls@getcounterprefix
4634 \fi

```

De-tok label if required

```

4635 \edef\@gls@label{\glsdetoklabel{#1}}%

```

Write the information to file:

```

4636 \@do@wrglossary
4637 }

```

\@do@wrglossary

```

4638 \newcommand*{\@do@wrglossary}{%

```

Determine whether to use xindy or makeindex syntax

```

4639 \ifglsxindy

```

Need to determine if the formatting information starts with a ( or ) indicating a range.

```

4640 \expandafter\@glo@check@mkidxrangechar\@gls@numberformat\@nil
4641 \def\@glo@range{}%
4642 \expandafter\if\@glo@prefix(\relax
4643 \def\@glo@range{:open-range}%
4644 \else
4645 \expandafter\if\@glo@prefix)\relax
4646 \def\@glo@range{:close-range}%
4647 \fi
4648 \fi

```

Write to the glossary file using xindy syntax.

```

4649 \glossary[\csname glo@\@gls@label @type\endcsname]{%
4650 (indexentry :tkey (\csname glo@\@gls@label @index\endcsname)
4651 :locref \string"\@glo@counterprefix}{\@gls@locref}\string" %
4652 :attr \string"\@gls@counter\@glo@suffix\string"
4653 \@glo@range
4654 )
4655 }%
4656 \else

```

Convert the format information into the format required for makeindex

```

4657 \@set@glo@numformat{\@glo@numfmt}{\@gls@counter}{\@gls@numberformat}}%
4658 {\@glo@counterprefix}%

```

Write to the glossary file using makeindex syntax.

```

4659 \glossary[\csname glo@\@gls@label @type\endcsname]{%
4660 \string\glossaryentry{\csname glo@\@gls@label @index\endcsname
4661 \@gls@encapchar\@glo@numfmt}{\@gls@locref}}%
4662 \fi
4663 }

```



`\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.)

```

4664 \newcommand*\@gls@getcounterprefix[2]{%
4665   \edef\@gls@thisloc{#1}\edef\@gls@thisHloc{#2}%
4666   \ifx\@gls@thisloc\@gls@thisHloc
4667     \def\@glo@counterprefix{}%
4668   \else
4669     \def\@gls@get@counterprefix##1.#1##2\end@getprefix{%
4670       \def\@glo@tmp{##2}%
4671       \ifx\@glo@tmp\@empty
4672         \def\@glo@counterprefix{}%
4673       \else
4674         \def\@glo@counterprefix{##1}%
4675       \fi
4676     }%
4677   \@gls@get@counterprefix#2.#1\end@getprefix

```

Warn if no prefix can be formed.

```

4678   \ifx\@glo@counterprefix\@empty
4679     \GlossariesWarning{Hyper target ‘#2’ can’t be formed by
4680       prefixing~Jlocation ‘#1’. You need to modify the
4681       definition of \string\theH\@gls@counter~Jotherwise you
4682       will get the warning: “‘name{\@gls@counter.#1}’ has been~J
4683       referenced but does not exist”}%
4684   \fi
4685 \fi
4686 }

```

## 1.14 Glossary Entry Cross-References

`\@do@seeglossary` Write the glossary entry with a cross reference. The first argument is the entry’s label, the second must be in the form `[\langle tag \rangle]{\langle list \rangle}`, where `\langle tag \rangle` is a tag such as “see” and `\langle list \rangle` is a list of labels.

```

4687 \newcommand{\@do@seeglossary}[2]{%
4688   \def\@gls@xref{#2}%
4689   \@onelevel@sanitize\@gls@xref
4690   \@gls@checkmkidxchars\@gls@xref
4691   \ifglxsindy
4692     \glossary[\csname glo@#1@type\endcsname]{%
4693       (indexentry
4694         :tkey (\csname glo@#1@index\endcsname)
4695         :xref (\string"\@gls@xref\string")
4696         :attr \string"see\string"
4697       )
4698     }%

```

```

4699 \else
4700   \glossary[\csname glo@#1@type\endcsname]{%
4701   \string\glossaryentry{\csname glo@#1@index\endcsname
4702   \@gls@encapchar glsseeformat\@gls@xref}{Z}}}%
4703 \fi
4704 }

```

`\@gls@fixbraces` If no optional argument is specified, list needs to be enclosed in a set of braces.

```

4705 \def\@gls@fixbraces#1#2#3\@nil{%
4706   \ifx#2[\relax
4707     \@gls@fixbraces#1#2#3\@end@fixbraces
4708   \else
4709     \def#1{{#2#3}}}%
4710   \fi
4711 }

```

`\@@gls@fixbraces`

```

4712 \def\@@gls@fixbraces#1[#2]#3\@end@fixbraces{%
4713   \def#1{[#2]{#3}}}%
4714 }

```

`\glssee` `\glssee{<label>}{<cross-ref list>}`

```

4715 \DeclareRobustCommand*\glssee}[3][\seename]{%
4716   \@do@seeglossary{#2}{#1}{#3}}%
4717 \newcommand*\@glssee}[3][\seename]{%
4718   \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.

```

4719 \DeclareRobustCommand*\glsseeformat}[3][\seename]{%
4720   \emph{#1} \glsseelist{#2}}

```

`\glsseelist` `\glsseelist{<list>}` formats list of entry labels.

```

4721 \DeclareRobustCommand*\glsseelist}[1]{%

```

If there is only one item in the list, set the last separator to do nothing.

```

4722   \let\@gls@dolast\relax

```

Don’t display separator on the first iteration of the loop

```

4723   \let\@gls@donext\relax

```

Iterate through the labels

```

4724   \@for\@gls@thislabel:=#1\do{%

```

Check if on last iteration of loop

```

4725     \ifx\@xfor@nextelement\@nnil

```

```

4726       \@gls@dolast

```

```

4727     \else

```

```

4728       \@gls@donext

```

```

4729     \fi

```

Display the entry for this label. (Expanding label as it's a temporary control sequence that's used elsewhere.)

```
4730 \expandafter\glsseeitem\expandafter{\@gls@thislabel}%
```

Update separators

```
4731 \let\@gls@dolast\glsseelastsep
```

```
4732 \let\@gls@donext\glsseesep
```

```
4733 }%
```

```
4734 }
```

`\glsseelastsep` Separator to use between penultimate and ultimate entries in a cross-referencing list.

```
4735 \newcommand*{\glsseelastsep}{\space\andname\space}
```

`\glsseesep` Separator to use between entries in a cross-referencing list.

```
4736 \newcommand*{\glsseesep}{, }
```

`\glsseeitem` `\glsseeitem{<label>}` formats individual entry in a cross-referencing list.

```
4737 \DeclareRobustCommand*{\glsseeitem}[1]{\gls hyperlink[\glsseeitemformat{#1}]{#1}}
```

`\glsseeitemformat` As from v3.0, default is to use `\glsentrytext` instead of `\glsentryname`. (To avoid problems with the name key being sanitized.)

```
4738 \newcommand*{\glsseeitemformat}[1]{\glsentrytext{#1}}
```

## 1.15 Displaying the glossary

An individual glossary is displayed in the text using `\printglossary[<key-val list>]`. If the type key is omitted, the default glossary is displayed. The optional argument can be used to specify an alternative glossary, and can also be used to set the style, title and entry in the table of contents. Available keys are defined below.

`\gls@save@numberlist` Provide command to store number list.

```
4739 \newcommand*{\gls@save@numberlist}[1]{%
```

```
4740 \ifglssavenumberlist
```

```
4741 \toks@{#1}%
```

```
4742 \edef\@do@writeaux@info{%
```

```
4743 \noexpand\csgdef{glo@\glscurrententrylabel @numberlist}{\the\toks@}%
```

```
4744 }%
```

```
4745 \@onelevel@sanitize\@do@writeaux@info
```

```
4746 \protected@write\@auxout{}\@do@writeaux@info}%
```

```
4747 \fi
```

```
4748 }
```

`\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.)

```
4749 \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.

```
4750 \ifcsundef{printglossary}{}%
```

```
4751 {%
```

If `\printglossary` is already defined, issue a warning and undefine it.

```
4752 \@gls@warnonglossdefined
```

```
4753 \undef\printglossary
```

```
4754 }
```

`\printglossary` has an optional argument. The default value is to set the glossary type to the main glossary.

```
4755 \newcommand*{\printglossary}[1][type=\glsdefaulttype]{%
```

```
4756 \@printglossary{#1}{\@print@glossary}%
```

```
4757 }
```

The `\printglossaries` command will do `\printglossary` for each glossary type that has been defined. It is better to use `\printglossaries` rather than individual `\printglossary` commands to ensure that you don't forget any new glossaries you may have created. It also makes it easier to chop and change the value of the acronym package option. However, if you want to list the glossaries in a different order, or if you want to set the title or table of contents entry, or if you want to use different glossary styles for each glossary, you will need to use `\printglossary` explicitly for each glossary type.

`\printglossaries`

```
4758 \newcommand*{\printglossaries}{%
```

```
4759 \foralllglossaries{\@glo@type}{\printglossary[type=\@glo@type]}}%
```

```
4760 }
```

`\printnoidxglossary` Provide an alternative to `\printglossary` that doesn't require an external indexing application. Entries won't be sorted and the location list will be empty.

```
4761 \newcommand*{\printnoidxglossary}[1][type=\glsdefaulttype]{%
```

```
4762 \@printglossary{#1}{\@print@noidx@glossary}%
```

```
4763 }
```

`\printnoidxglossaries` Analogous to `\printglossaries`

```
4764 \newcommand*{\printnoidxglossaries}{%
```

```
4765 \foralllglossaries{\@glo@type}{\printnoidxglossary[type=\@glo@type]}}%
```

```
4766 }
```

`@printgloss@setsort` Initialise to do nothing.

```
4767 \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.

```

4768 \newcommand{\@printglossary}[2]{%
    Set up defaults.
4769   \def\@glo@type{\glsdefaulttype}%
4770   \def\glossarytitle{\csname @glo@type\@glo@type @title\endcsname}%

4771   \def\glossarytoctitle{\glossarytitle}%
4772   \let\org@glossarytitle\glossarytitle
4773   \def\@glossarystyle{}%
4774   \def\gls@dotoc@title{\glssettoc@title{\@glo@type}}%

    Store current value of \glossaryentrynumbers. (This may be changed via the
    optional argument)
4775   \let\org@glossaryentrynumbers\glossaryentrynumbers

    Localise the effects of the optional argument
4776   \bgroup

    Activate or deactivate sort key:
4777   \@printgloss@setsort

    Determine settings specified in the optional argument.
4778   \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)
4779   \ifx\glossarytitle\org@glossarytitle
4780   \else
4781     \expandafter\let\csname @glo@type\@glo@type @title\endcsname
4782     \glossarytitle
4783   \fi

    Allow a high-level user command to indicate the current glossary
4784   \let\currentglossary\@glo@type

    Enable individual number lists to be suppressed.
4785   \let\org@glossaryentrynumbers\glossaryentrynumbers
4786   \let\glsnonextpages\@glsnonextpages

    Enable individual number list to be activated:
4787   \let\glsnextpages\@glsnextpages

    Enable suppression of description terminators.
4788   \let\nopostdesc\@nopostdesc

    Set up the entry for the TOC
4789   \gls@dotoc@title

    Set the glossary style
4790   \@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):

```

4791 \let\gls@org@glossaryentryfield\glossentry
4792 \let\gls@org@glossarysubentryfield\subglossentry
4793 \renewcommand{\glossentry}[1]{%
4794   \xdef\glscurrententrylabel{\glsdetoklabel{##1}}%
4795   \gls@org@glossaryentryfield{##1}%
4796 }%
4797 \renewcommand{\subglossentry}[2]{%
4798   \xdef\glscurrententrylabel{\glsdetoklabel{##2}}%
4799   \gls@org@glossarysubentryfield{##1}{##2}%
4800 }%

```

Now do the handler macro that deals with the actual glossary:

```

4801 #2%

End the current scope
4802 \egroup

Reset \glossaryentrynumbers
4803 \global\let\glossaryentrynumbers\@org@glossaryentrynumbers

Suppress warning about no \printglossary
4804 \global\let\warn@noprintglossary\relax
4805 }

```

`\@print@glossary` Internal workings of `\printglossary` dealing with reading the external file.

```

4806 \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.)

```

4807 \makeatletter

```

Input the glossary file, if it exists.

```

4808 \@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.

```

4809 \IfFileExists{\jobname.\csname @glotype@\@glo@type @in\endcsname}%
4810 {}%
4811 {\null}%

```

If `xindy` is being used, need to write the language dependent information to the `.aux` file for `makeglossaries`.

```

4812 \ifglxindy
4813   \ifcsundef{@xdy@\@glo@type @language}%
4814   {%
4815     \edef\@do@auxoutstuff{%
4816       \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.

```

4817         \noexpand\immediate\noexpand\write\@auxout{%
4818         \string\providecommand\string\@xdylanguage[2]{}%
4819         \noexpand\immediate\noexpand\write\@auxout{%
4820         \string\@xdylanguage{\@glo@type}{\@xdy@main@language}}%
4821     }%
4822 }%
4823 }%
4824 {%
4825     \edef\@do@auxoutstuff{%
4826     \noexpand\AtEndDocument{%
4827     \noexpand\immediate\noexpand\write\@auxout{%
4828     \string\providecommand\string\@xdylanguage[2]{}%
4829     \noexpand\immediate\noexpand\write\@auxout{%
4830     \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type
4831     @language\endcsname}}%
4832     }%
4833     }%
4834     }%
4835     \@do@auxoutstuff
4836     \edef\@do@auxoutstuff{%
4837     \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.

```

4838         \noexpand\immediate\noexpand\write\@auxout{%
4839         \string\providecommand\string\@gls@codepage[2]{}%
4840         \noexpand\immediate\noexpand\write\@auxout{%
4841         \string\@gls@codepage{\@glo@type}{\gls@codepage}}%
4842     }%
4843 }%
4844 \@do@auxoutstuff
4845 \fi

```

Activate warning if \makeglossaries hasn't been used.

```

4846 \renewcommand*{\@warn@nomakeglossaries}{%
4847 \GlossariesWarningNoLine{\string\makeglossaries\space
4848 hasn't been used,^^Jthe glossaries will not be updated}%
4849 }%
4850 }

```

The sort macros all have the syntax:

`\@glo@sortmacro@<order>{<type>}`

where  $\langle order \rangle$  is the sort order as specified by the sort key and  $\langle type \rangle$  is the glossary type. (The referenced entry list is stored in  $\backslash@glsref@{\langle type \rangle}$ . The actual sorting is done by  $\backslash@glo@sortentries{\langle handler \rangle}{\langle type \rangle}$ ).

$\backslash@glo@sortentries$

```

4851 \newcommand*{\@glo@sortentries}[2]{%
4852   \def\@glo@sortinglist{}%
4853   \def\@glo@sortinghandler{#1}%
4854   \edef\@glo@type{#2}%
4855   \forlistcsloop{\@glo@do@sortentries}{\@glsref@#2}%
4856   \csdef{\@glsref@#2}{}%
4857   \@for\@this@label:=\@glo@sortinglist\do{%

```

Has this entry already been added?

```

4858     \xifinlistcs{\@this@label}{\@glsref@#2}%
4859     {}%
4860     {%
4861       \listcsxadd{\@glsref@#2}{\@this@label}%
4862     }%
4863     \ifcsdef{\@glo@sortingchildren@\@this@label}%
4864     {%
4865       \@glo@addchildren{#2}{\@this@label}%
4866     }%
4867     {}%
4868   }%
4869 }

```

$\backslash@glo@addchildren$   $\backslash@glo@addchildren{\langle type \rangle}{\langle parent \rangle}$

```

4870 \newcommand*{\@glo@addchildren}[2]{%

```

Scope to allow nesting.

```

4871   \bgroup
4872   \letcs{\@glo@childlist}{\@glo@sortingchildren@#2}%
4873   \@for\@this@childlabel:=\@glo@childlist\do
4874   {%

```

Check this label hasn't already been added.

```

4875     \xifinlistcs{\@this@childlabel}{\@glsref@#1}%
4876     {}%
4877     {%
4878       \listcsxadd{\@glsref@#1}{\@this@childlabel}%
4879     }%

```

Does this child have children?

```

4880     \ifcsdef{\@glo@sortingchildren@\@this@childlabel}%
4881     {%
4882       \@glo@addchildren{#1}{\@this@childlabel}%
4883     }%

```



```

4884      {%
4885      }%
4886      }%
4887      \egroup
4888 }

```

@glo@do@sortentries

```

4889 \newcommand*{\@glo@do@sortentries}[1]{%
4890   \ifglshasparent{#1}%
4891   {%
      This entry has a parent, so add it to the child list
4892   \edef\@glo@parent{\csuse{glo@\glstetoklabel{#1}@parent}}%
4893   \ifcsundef{\@glo@sortingchildren@\@glo@parent}%
4894   {%
4895     \csdef{\@glo@sortingchildren@\@glo@parent}{}%
4896   }%
4897   }%
4898   \expandafter\@glo@sortedinsert
4899   \csname @glo@sortingchildren@\@glo@parent\endcsname{#1}%

```

Has the parent been added?

```

4900   \xifinlistcs{\@glo@parent}{\glstref@\@glo@type}%
4901   {%

```

Yes, it has so do nothing.

```

4902   }%
4903   {%

```

No, it hasn't so add it now.

```

4904     \expandafter\@glo@do@sortentries\expandafter{\@glo@parent}%
4905   }%
4906 }%
4907 {%
4908   \@glo@sortedinsert{\@glo@sortinglist}{#1}%
4909 }%
4910 }

```

\@glo@sortedinsert      \@glo@sortedinsert{<list>}{<entry label>}

Insert into list.

```

4911 \newcommand*{\@glo@sortedinsert}[2]{%
4912   \dtlinsertinto{#2}{#1}{\@glo@sortinghandler}%
4913 }%

```

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

```
4914 \newcommand*{\@glo@sorthandler@word}[2]{%
4915   \letcs\@gls@sort@A{glo@\glsdetoklabel{#1}@sort}%
4916   \letcs\@gls@sort@B{glo@\glsdetoklabel{#2}@sort}%
4917   \edef\glo@do@compare{%
4918     \noexpand\dtlwordindexcompare{\noexpand\dtl@sortresult}%
4919     {\expandonce\@gls@sort@B}%
4920     {\expandonce\@gls@sort@A}%
4921   }%
4922   \glo@do@compare
4923 }
```

@sorthandler@letter

```
4924 \newcommand*{\@glo@sorthandler@letter}[2]{%
4925   \letcs\@gls@sort@A{glo@\glsdetoklabel{#1}@sort}%
4926   \letcs\@gls@sort@B{glo@\glsdetoklabel{#2}@sort}%
4927   \edef\glo@do@compare{%
4928     \noexpand\dtlletterindexcompare{\noexpand\dtl@sortresult}%
4929     {\expandonce\@gls@sort@B}%
4930     {\expandonce\@gls@sort@A}%
4931   }%
4932   \glo@do@compare
4933 }
```

lo@sorthandler@case Case-sensitive sort.

```
4934 \newcommand*{\@glo@sorthandler@case}[2]{%
4935   \letcs\@gls@sort@A{glo@\glsdetoklabel{#1}@sort}%
4936   \letcs\@gls@sort@B{glo@\glsdetoklabel{#2}@sort}%
4937   \edef\glo@do@compare{%
4938     \noexpand\dtlcompare{\noexpand\dtl@sortresult}%
4939     {\expandonce\@gls@sort@B}%
4940     {\expandonce\@gls@sort@A}%
4941   }%
4942   \glo@do@compare
4943 }
```

@sorthandler@nocase Case-insensitive sort.

```
4944 \newcommand*{\@glo@sorthandler@nocase}[2]{%
4945   \letcs\@gls@sort@A{glo@\glsdetoklabel{#1}@sort}%
4946   \letcs\@gls@sort@B{glo@\glsdetoklabel{#2}@sort}%
4947   \edef\glo@do@compare{%
4948     \noexpand\dtlicompare{\noexpand\dtl@sortresult}%
4949     {\expandonce\@gls@sort@B}%
4950     {\expandonce\@gls@sort@A}%
4951   }%
4952   \glo@do@compare
4953 }
```

@glo@sortmacro@word Sort macro for ‘word’

```
4954 \newcommand*{\@glo@sortmacro@word}[1]{%
4955   \ifdefstring{\@glo@default@sorttype}{standard}%
4956   {%
4957     \@glo@sortentries{\@glo@sorthandler@word}{#1}%
4958   }%
4959   {%
4960     \PackageError{glossaries}{Conflicting sort options:^^J
4961       \string\usepackage[sort=\@glo@default@sorttype]{glossaries}^^J
4962       \string\printnoidxglossary[sort=word]}{}%
4963   }%
4964 }
```

@lo@sortmacro@letter Sort macro for ‘letter’

```
4965 \newcommand*{\@glo@sortmacro@letter}[1]{%
4966   \ifdefstring{\@glo@default@sorttype}{standard}%
4967   {%
4968     \@glo@sortentries{\@glo@sorthandler@letter}{#1}%
4969   }%
4970   {%
4971     \PackageError{glossaries}{Conflicting sort options:^^J
4972       \string\usepackage[sort=\@glo@default@sorttype]{glossaries}^^J
4973       \string\printnoidxglossary[sort=letter]}{}%
4974   }%
4975 }
```

@sortmacro@standard Sort macro for ‘standard’. (Use either ‘word’ or ‘letter’ order.)

```
4976 \newcommand*{\@glo@sortmacro@standard}[1]{%
4977   \ifdefstring{\@glo@default@sorttype}{standard}%
4978   {%
4979     \ifcsdef{\@glo@sorthandler@\glsorder}%
4980     {%
4981       \@glo@sortentries{\csuse{\@glo@sorthandler@\glsorder}}{#1}%
4982     }%
4983     {%
4984       \PackageError{glossaries}{Unknown sort handler ‘\glsorder’}{}%
4985     }%
4986   }%
4987   {%
4988     \PackageError{glossaries}{Conflicting sort options:^^J
4989       \string\usepackage[sort=\@glo@default@sorttype]{glossaries}^^J
4990       \string\printnoidxglossary[sort=standard]}{}%
4991   }%
4992 }
```

@glo@sortmacro@case Sort macro for ‘case’

```
4993 \newcommand*{\@glo@sortmacro@case}[1]{%
4994   \ifdefstring{\@glo@default@sorttype}{standard}%
4995   {%
```

```

4996     \@glo@sortentries{\@glo@sorthandler@case}{#1}%
4997 }%
4998 {%
4999     \PackageError{glossaries}{Conflicting sort options:^^J
5000     \string\usepackage[sort=\@glo@default@sorttype]{glossaries}^^J
5001     \string\printnoidxglossary[sort=case]}{%}%
5002 }%
5003 }

\lo@sortmacro@nocase Sort macro for 'nocase'

5004 \newcommand*{\@glo@sortmacro@nocase}[1]{%
5005     \ifdefstring{\@glo@default@sorttype}{standard}%
5006     {%
5007         \@glo@sortentries{\@glo@sorthandler@nocase}{#1}%
5008     }%
5009     {%
5010         \PackageError{glossaries}{Conflicting sort options:^^J
5011         \string\usepackage[sort=\@glo@default@sorttype]{glossaries}^^J
5012         \string\printnoidxglossary[sort=nocase]}{%}%
5013     }%
5014 }

\@glo@sortmacro@def Sort macro for 'def'. The order of definition is given in \glolist@<type>.

5015 \newcommand*{\@glo@sortmacro@def}[1]{%
5016     \def\@glo@sortinglist{}%
5017     \forlslentries[#1]{\@gls@thislabel}%
5018     {%
5019         \xifinlistcs{\@gls@thislabel}{\@glsref@#1}%
5020         {%
5021             \listcsadd{\@glo@sortinglist}{\@gls@thislabel}%
5022         }%
5023         {%
5024             }%
5025         }%
5026         \cslet{\@glsref@#1}{\@glo@sortinglist}%
5027     }

\lo@sortmacro@def@do This won't include parent entries that haven't been referenced.

5028 \newcommand*{\@glo@sortmacro@def@do}[1]{%
5029     \ifinlistcs{#1}{\@glsref@\@glo@type}%
5030     {}%
5031     {%
5032         \listcsadd{\@glsref@\@glo@type}{#1}%
5033     }%
5034     \ifcsdef{\@glo@sortingchildren@#1}%
5035     {%
5036         \@glo@addchildren{\@glo@type}{#1}%

```

```

5037 }%
5038 {}%
5039 }

\@glo@sortmacro@use Sort macro for 'use'. (No sorting is required, as the entries are already in order
of use, so do nothing.)
5040 \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.
5041 \newcommand*{\@print@noidx@glossary}{%
5042 \ifcsdef{@glsref@ \@glo@type}%
5043 {%
Sort the entries:
5044 \ifcsdef{@glo@sortmacro@ \@glo@sorttype}%
5045 {%
5046 \csuse{@glo@sortmacro@ \@glo@sorttype}{\@glo@type}%
5047 }%
5048 {%
5049 \PackageError{glossaries}{Unknown sort handler '@@glo@sorttype'}{ }%
5050 }%

Do the glossary heading and preamble
5051 \glossarysection[\glossarytoctitle]{\glossarytitle}%
5052 \glossarypreamble
5053 \begin{theglossary}%
5054 \glossaryheader
5055 \glsresetentrylist
5056 \def\@gls@currentlettergroup{}%

Iterate through the entries.
5057 \forlistcsloop{\@gls@noidx@do}{\@glsref@ \@glo@type}%

Finally end the glossary and do the postamble:
5058 \end{theglossary}%
5059 \glossarypostamble
5060 }%
5061 {%
5062 \@gls@noref@warn{\@glo@type}%
5063 }%
5064 }

\glo@grabfirst
5065 \def\glo@grabfirst#1#2\@nil{%
5066 \def\@gls@firsttok{#1}%

```

```

5067 \ifdefempty\@gls@firsttok
5068 {%
5069 \def\@glo@thislettergrp{0}%
5070 }%
5071 {%

Sanitize it:
5072 \@onelevel@sanitize\@gls@firsttok

Fetch the first letter:
5073 \expandafter\@glo@grabfirst\@gls@firsttok{}\}\@nil
5074 }%
5075 }

```

\@glo@grabfirst

```

5076 \def\@glo@grabfirst#1#2\@nil{%
5077 \ifdefempty\@glo@thislettergrp
5078 {%
5079 \def\@glo@thislettergrp{glssymbols}%
5080 }%
5081 {%
5082 \count@=\uccode'#1\relax
5083 \ifnum\count@=0\relax
5084 \def\@glo@thislettergrp{glssymbols}%
5085 \else
5086 \ifdefstring\@glo@sorttype{case}%
5087 {%
5088 \count@='#1\relax
5089 }%
5090 {%
5091 }%
5092 \edef\@glo@thislettergrp{\the\count@}%
5093 \fi
5094 }%
5095 }

```

\@gls@noidx@do Handler for list iteration used by \@print@noidx@glossary. The argument is the entry label. This only allows one sublevel.

```

5096 \newcommand{\@gls@noidx@do}[1]{%

Get this entry's location list
5097 \global\letcs{\@gls@loclist}{glo\@glsdetoklabel{#1}@loclist}%

Does this entry have a parent?
5098 \ifglshasparent{#1}%
5099 {%

Has a parent.
5100 \gls@level=\csuse{glo\@glsdetoklabel{#1}@level}\relax
5101 \ifdefvoid{\@gls@loclist}
5102 {%

```

```

5103     \subglossentry{\gls@level}{#1}{}%
5104 }%
5105 {%
5106     \subglossentry{\gls@level}{#1}%
5107     {%
5108         \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
5109     }%
5110 }%
5111 }%
5112 {%

```

Doesn't have a parent Get this entry's sort key

```

5113     \letcs{\@gls@sort}{glo@glsdetoklabel{#1}@sort}%

```

Fetch the first letter:

```

5114     \expandafter\glo@grabfirst\@gls@sort{}{}\@nil
5115     \ifdefequal{\@glo@thislettergrp}{\@gls@currentlettergroup}%
5116     {}%
5117     {%

```

Do the group header:

```

5118     \ifdefempty{\@gls@currentlettergroup}{\@gls@groupskip}%
5119     \gls@groupheading{\@glo@thislettergrp}%
5120 }%
5121 \let\@gls@currentlettergroup\@glo@thislettergrp

```

Do this entry:

```

5122     \ifdefvoid{\@gls@loclist}
5123     {%
5124         \glossentry{#1}{}%
5125     }%
5126     {%
5127         \glossentry{#1}%
5128     }%
5129     \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
5130 }%
5131 }%
5132 }%
5133 }

```

\glsnoidxloclist `\glsnoidxloclist{<list cs>}`

Display location list.

```

5134 \newcommand*{\glsnoidxloclist}[1]{%
5135     \def\@gls@noidxloclist@sep{}%
5136     \def\@gls@noidxloclist@prev{}%
5137     \forlistloop{\glsnoidxloclisthandler}{#1}%
5138 }

```

noidxloclisthandler    Handler for location list iterator.

```
5139 \newcommand*{\glsnoidxloclisthandler}[1]{%
5140   \ifdefstring{\@gls@noidxloclist@prev}{#1}%
5141   {%
      Same as previous location so skip.
5142   }%
5143   {%
5144     \@gls@noidxloclist@sep
5145     #1%
5146     \def\@gls@noidxloclist@sep{\delimN}%
5147     \def\@gls@noidxloclist@prev{#1}%
5148   }%
5149 }
```

splayloclisthandler    Handler for location list iterator when used with \glsdisplaynumberlist.

```
5150 \newcommand*{\glsnoidxdisplayloclisthandler}[1]{%
5151   \ifdefstring{\@gls@noidxloclist@prev}{#1}%
5152   {%
      Same as previous location so skip.
5153   }%
5154   {%
5155     \@gls@noidxloclist@sep
5156     \@gls@noidxloclist@prev
5157     \def\@gls@noidxloclist@prev{#1}%
5158   }%
5159 }
```

\glsnoidxdisplayloc    `\glsnoidxdisplayloc{<prefix>}{<counter>}{<format>}{<location>}`

Display a location in the location list.

```
5160 \newcommand*\glsnoidxdisplayloc[4]{%
5161   \setentrycounter[#1]{#2}%
5162   \csuse{#3}{#4}%
5163 }
```

\@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.

```
5164 \newcommand*\@gls@reference[3]{%
      Add to label list
5165   \glsdoifexistsorwarn{#2}%
5166   {%
5167     \ifcsundef{\@glsref@#1}{\csgdef{\@glsref@#1}{}}{}%

```



```

5168 \ifinlistcs{#2}{@glsref@#1}%
5169 {}%
5170 {\listcsgadd{@glsref@#1}{#2}}%
    Add to location list
5171 \ifcsundef{glo@glstdetoklabel{#2}@loclist}%
5172 {\csgdef{glo@glstdetoklabel{#2}@loclist}{}}%
5173 {}%
5174 \listcsgadd{glo@glstdetoklabel{#2}@loclist}{#3}%
5175 }%
5176 }

```

The keys that can be used in the optional argument to `\printglossary` or `\printnoidxglossary` are as follows: The `type` key sets the glossary type.

```

5177 \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`.

```

5178 \define@key{printgloss}{title}{%
5179 \def@glossarytitle{#1}%
5180 \let@gls@dotoc@title\relax
5181 }

```

The `toctitle` sets the text used for the relevant entry in the table of contents.

```

5182 \define@key{printgloss}{toctitle}{%
5183 \def@glossarytoctitle{#1}%
5184 \let@gls@dotoc@title\relax
5185 }

```

The `style` key sets the glossary style (but only for the given glossary).

```

5186 \define@key{printgloss}{style}{%
5187 \ifcsundef{@glsstyle@#1}%
5188 {%
5189 \PackageError{glossaries}%
5190 {Glossary style ‘#1’ undefined}{}%
5191 }%
5192 {%
5193 \def@glossarystyle{\setglossentrycompatibility
5194 \csname @glsstyle@#1\endcsname}%
5195 }%
5196 }

```

The `numberedsection` key determines if this glossary should be in a numbered section.

```

5197 \define@choicekey{printgloss}{numberedsection}[\val\nr]{%
5198 false,nolabel,autolabel,nameref}[nolabel]{%
5199 \ifcase\nr\relax
5200 \renewcommand*{\@@glossarysecstar}{*}%
5201 \renewcommand*{\@@glossaryseclabel}{}%
5202 \or
5203 \renewcommand*{\@@glossarysecstar}{}%

```

```

5204 \renewcommand*{\@@glossaryseclabel}{}%
5205 \or
5206 \renewcommand*{\@@glossarysecstar}{}%
5207 \renewcommand*{\@@glossaryseclabel}{\label{\glsautoprefix\@glo@type}}%
5208 \or
5209 \renewcommand*{\@@glossarysecstar}{*}%
5210 \renewcommand*{\@@glossaryseclabel}{%
5211 \protected@edef\@currentlabelname{\glossarytoctitle}%
5212 \label{\glsautoprefix\@glo@type}}%
5213 \fi
5214 }

```

The `nogroupskip` key determines whether or not there should be a vertical gap between glossary groups.

```

5215 \define@choicekey{printgloss}{nogroupskip}{true,false}[true]{%
5216 \csuse{glsnogroupskip#1}%
5217 }

```

The `nopostdot` key has the same effect as the package option of the same name.

```

5218 \define@choicekey{printgloss}{nopostdot}{true,false}[true]{%
5219 \csuse{glsnopostdot#1}%
5220 }

```

The `entrycounter` key is the same as the package option but localised to the current glossary.

```

5221 \define@choicekey{printgloss}{entrycounter}{true,false}[true]{%
5222 \csuse{glsentrycounter#1}%
5223 \ifglsentrycounter
5224 \ifx\@gls@counterwithin\@empty
5225 \newcounter{glossaryentry}%
5226 \else
5227 \newcounter{glossaryentry}[\@gls@counterwithin]%
5228 \fi
5229 \def\theHglossaryentry{\currentglossary.\theglossaryentry}%
5230 \renewcommand*{\glsresetentrycounter}{%
5231 \setcounter{glossaryentry}{0}%
5232 }%
5233 \renewcommand*{\glsstepentry}[1]{%
5234 \refstepcounter{glossaryentry}%
5235 \label{glsentry-\glsdetoklabel{##1}}%
5236 }%
5237 \renewcommand*{\glsentrycounterlabel}{\theglossaryentry.\space}%
5238 \renewcommand*{\glsentryitem}[1]{%
5239 \glsstepentry{##1}\glsentrycounterlabel
5240 }%
5241 \else
5242 \renewcommand*{\glsresetentrycounter}{}%
5243 \renewcommand*{\glsstepentry}[1]{%
5244 \renewcommand*{\glsentrycounterlabel}{}%

```

```

5245 \renewcommand*{\glsubentryitem}[1]{\glresetsubentrycounter}
5246 \fi
5247 }

```

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.

```

5248 \define@choicekey{printgloss}{subentrycounter}{true,false}[true]{%
5249 \csuse{glssubentrycounter#1}%
5250 \ifglssubentrycounter
5251 \ifundef\c@glossarysubentry
5252 {%
5253 \ifglsubentrycounter
5254 \newcounter{glossarysubentry}[glossaryentry]%
5255 \else
5256 \newcounter{glossarysubentry}
5257 \fi
5258 }{}%
5259 \renewcommand*{\glssubentry}[1]{%
5260 \edef\currentglssubentry{\glsetoklabel{##1}}%
5261 \refstepcounter{glossarysubentry}%
5262 \label{glsubentry-\currentglssubentry}%
5263 }%
5264 \renewcommand*{\glresetsubentrycounter}{%
5265 \setcounter{glossarysubentry}{0}%
5266 }%
5267 \renewcommand*{\glssubentryitem}[1]{%
5268 \glssubentry{##1}\glssubentrycounterlabel
5269 }%
5270 \renewcommand*{\glssubentrycounterlabel}{\theglossarysubentry}\space}%
5271 \def\theHglossarysubentry{\currentglssubentry.\theglossarysubentry}
5272 \else
5273 \renewcommand*{\glssubentryitem}[1]{}%
5274 \renewcommand*{\glssubentry}[1]{}%
5275 \renewcommand*{\glresetsubentrycounter}{}%
5276 \renewcommand*{\glssubentrycounterlabel}{}%
5277 \fi
5278 }

```

The nonumberlist key determines if this glossary should have a number list.

```

5279 \define@boolkey{printgloss}[gls]{nonumberlist}[true]{%
5280 \ifglsnonumberlist
5281 \def\glossaryentrynumbers##1{}%
5282 \else
5283 \def\glossaryentrynumbers##1{##1}%
5284 \fi}

```

The sort key sets the glossary sort handler (\printnoidxglossary only).

```

5285 \define@key{printgloss}{sort}{\@glo@assign@sortkey{#1}}

```

`\@no@assign@sortkey` Issue error if used with `\printglossary`

```

5286 \newcommand*{\@glo@no@assign@sortkey}[1]{%
5287   \PackageError{glossaries}{‘sort’ key not permitted with
5288   \string\printglossary}%
5289   {The ‘sort’ key may only be used with \string\printnoidxglossary}%
5290 }
```

`\@glo@assign@sortkey` For use with `\printnoidxglossary`

```

5291 \newcommand*{\@glo@assign@sortkey}[1]{%
5292   \def\@glo@sorttype{#1}%
5293 }
```

`\@glsnonetpages` 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 `\glsnonetpages` 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.

```

5294 \newcommand*{\@glsnonetpages}{%
5295   \gdef\glossaryentrynumbers##1{%
5296     \glsresetentrylist
5297   }%
5298 }
```

`\@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.

```

5299 \newcommand*{\@glsnextpages}{%
5300   \gdef\glossaryentrynumbers##1{%
5301     ##1\glsresetentrylist}}

```

`\glsresetentrylist` Resets `\glossaryentrynumbers`

```

5302 \newcommand*{\glsresetentrylist}{%
5303   \global\let\glossaryentrynumbers\org@glossaryentrynumbers}

```

`\glsnonetpages` Outside of `\printglossary` this does nothing.

```

5304 \newcommand*{\glsnonetpages}{}

```

`\glsnextpages` Outside of `\printglossary` this does nothing.

```

5305 \newcommand*{\glsnextpages}{}

```

`glossaryentry` If the `entrycounter` package option has been used, define a counter to number each level 0 entry.

```

5306 \ifglsetentrycounter
5307   \ifx\@gls@counterwithin\@empty
5308     \newcounter{glossaryentry}

```

```

5309 \else
5310 \newcounter{glossaryentry}[\@gls@counterwithin]
5311 \fi
5312 \def\theHglossaryentry{\currentglossary.\theglossaryentry}
5313 \fi

```

**glossarysubentry** If the subentrycounter package option has been used, define a counter to number each level 1 entry.

```

5314 \ifglssubentrycounter
5315 \ifglentrycounter
5316 \newcounter{glossarysubentry}[glossaryentry]
5317 \else
5318 \newcounter{glossarysubentry}
5319 \fi
5320 \def\theHglossarysubentry{\currentglssubentry.\theglossarysubentry}
5321 \fi

```

**resetsubentrycounter** Resets the glossarysubentry counter.

```

5322 \ifglssubentrycounter
5323 \newcommand*{\glsresetsubentrycounter}{%
5324 \setcounter{glossarysubentry}{0}%
5325 }
5326 \else
5327 \newcommand*{\glsresetsubentrycounter}{}
5328 \fi

```

**resetentrycounter** Resets the glossaryentry counter.

```

5329 \ifglentrycounter
5330 \newcommand*{\glsresetentrycounter}{%
5331 \setcounter{glossaryentry}{0}%
5332 }
5333 \else
5334 \newcommand*{\glsresetentrycounter}{}
5335 \fi

```

**\glsstepentry** Advance the glossaryentry counter if in use. The argument is the label associated with the entry.

```

5336 \ifglentrycounter
5337 \newcommand*{\glsstepentry}[1]{%
5338 \refstepcounter{glossaryentry}%
5339 \label{glentry-\glsdetoklabel{#1}}%
5340 }
5341 \else
5342 \newcommand*{\glsstepentry}[1]{}
5343 \fi

```

**\glsstepsubentry** Advance the glossarysubentry counter if in use. The argument is the label associated with the subentry.

```

5344 \ifglssubentrycounter
5345   \newcommand*{\glssubentry}[1]{%
5346     \edef\currentglssubentry{\glsdetoklabel{#1}}%
5347     \refstepcounter{glossarysubentry}%
5348     \label{glentry-\currentglssubentry}%
5349   }
5350 \else
5351   \newcommand*{\glssubentry}[1]{%
5352 \fi

```

**\glssrefentry** Reference the entry or sub-entry counter if in use, otherwise just do `\gls`.

```

5353 \ifglentrycounter
5354   \newcommand*{\glssrefentry}[1]{\ref{glentry-\glsdetoklabel{#1}}}
5355 \else
5356   \ifglssubentrycounter
5357     \newcommand*{\glssrefentry}[1]{\ref{glentry-\glsdetoklabel{#1}}}
5358   \else
5359     \newcommand*{\glssrefentry}[1]{\gls{#1}}
5360   \fi
5361 \fi

```

**glssentrycounterlabel** Defines how to display the glossaryentry counter.

```

5362 \ifglentrycounter
5363   \newcommand*{\glssentrycounterlabel}{\theglossaryentry.\space}
5364 \else
5365   \newcommand*{\glssentrycounterlabel}{}
5366 \fi

```

**glssubentrycounterlabel** Defines how to display the glossarysubentry counter.

```

5367 \ifglssubentrycounter
5368   \newcommand*{\glssubentrycounterlabel}{\theglossarysubentry.\space}
5369 \else
5370   \newcommand*{\glssubentrycounterlabel}{}
5371 \fi

```

**\glssentryitem** Step and display glossaryentry counter, if appropriate.

```

5372 \ifglentrycounter
5373   \newcommand*{\glssentryitem}[1]{%
5374     \glssstepentry{#1}\glssentrycounterlabel
5375   }
5376 \else
5377   \newcommand*{\glssentryitem}[1]{\glssresetsubentrycounter}
5378 \fi

```

**\glsssubentryitem** Step and display glossarysubentry counter, if appropriate.

```

5379 \ifglssubentrycounter
5380   \newcommand*{\glsssubentryitem}[1]{%
5381     \glssstepsubentry{#1}\glsssubentrycounterlabel
5382   }

```

```

5383 \else
5384   \newcommand*{\glssubentryitem}[1]{%
5385 \fi

```

**theglossary** If the theglossary environment has already been defined, a warning will be issued. This environment should be redefined by glossary styles.

```

5386 \ifcsundef{theglossary}%
5387 {%
5388   \newenvironment{theglossary}{}{}%
5389 }%
5390 {%
5391   \@gls@warnontheglossdefined
5392   \renewenvironment{theglossary}{}{}%
5393 }

```

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`

```

5394 \newcommand*{\glossaryheader}{}

```

`\glstarget` `\glstarget{<label>}{<name>}`

Provide user interface to `\@glstarget` to make it easier to modify the glossary style in the document.

```

5395 \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>}`

```

5396 \providecommand*{\compatibleglossentry}[2]{%
5397   \toks@{#2}%
5398   \protected@edef\@do@glossentry{\noexpand\glossaryentryfield{#1}%
5399     {\noexpand\glsnamefont
5400       {\expandafter\expandonce\csname glo@#1@name\endcsname}}}%
5401     {\expandafter\expandonce\csname glo@#1@desc\endcsname}}%
5402     {\expandafter\expandonce\csname glo@#1@symbol\endcsname}}%
5403     {\the\toks@}%
5404   }%

```

```

5405 \do@glossentry
5406 }

```

\glossentryname

```

5407 \newcommand*{\glossentryname}[1]{%
5408   \glsdoifexistsorwarn{#1}%
5409   {%
5410     \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
5411     \expandafter\glsnamefont\expandafter{\glo@name}%
5412   }%
5413 }

```

\Glossentryname

```

5414 \newcommand*{\Glossentryname}[1]{%
5415   \glsdoifexistsorwarn{#1}%
5416   {%
5417     \glsnamefont{\Glsentryname{#1}}%
5418   }%
5419 }

```

\glossentrydesc

```

5420 \newcommand*{\glossentrydesc}[1]{%
5421   \glsdoifexistsorwarn{#1}%
5422   {%
5423     \glentrydesc{#1}%
5424   }%
5425 }

```

\Glossentrydesc

```

5426 \newcommand*{\Glossentrydesc}[1]{%
5427   \glsdoifexistsorwarn{#1}%
5428   {%
5429     \Glsentrydesc{#1}%
5430   }%
5431 }

```

\glossentrysymbol

```

5432 \newcommand*{\glossentrysymbol}[1]{%
5433   \glsdoifexistsorwarn{#1}%
5434   {%
5435     \glentrysymbol{#1}%
5436   }%
5437 }

```

\Glossentrysymbol

```

5438 \newcommand*{\Glossentrysymbol}[1]{%
5439   \glsdoifexistsorwarn{#1}%
5440   {%

```



```

5441 \Glsentrysymbol{#1}%
5442 }%
5443 }

```

compatiblesubglossentry `\subglossentry{<level>}{<label>}{<page-list>}`

```

5444 \providecommand*\compatiblesubglossentry}[3]{%
5445   \toks@{#3}%
5446   \protected@edef\@do@subglossentry{\noexpand\glossarysubentryfield{\number#1}%
5447     {#2}%
5448     {\noexpand\glsnamefont
5449       {\expandafter\expandonce\csname glo@#2@name\endcsname}}%
5450     {\expandafter\expandonce\csname glo@#2@desc\endcsname}%
5451     {\expandafter\expandonce\csname glo@#2@symbol\endcsname}%
5452     {\the\toks@}}%
5453   }%
5454   \@do@subglossentry
5455 }

```

sentrycompatibility

```

5456 \newcommand*\setglossentrycompatibility{%
5457   \let\glossentry\compatibleglossentry
5458   \let\subglossentry\compatiblesubglossentry
5459 }
5460 \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.

```

5461 \newcommand{\glossaryentryfield}[5]{%
5462   \GlossariesWarning
5463   {Deprecated use of \string\glossaryentryfield.^^J
5464     I recommend you change to \string\glossentry.^^J
5465     If you've just upgraded, try removing your gls auxiliary
5466     files^^J and recompile}%
5467   \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.)

```
5468 \newcommand*{\glossarysubentryfield}[6]{%
5469   \GlossariesWarning
5470   {Deprecated use of \string\glossarysubentryfield.^^J
5471     I recommend you change to \string\subglossentry.^^J
5472     If you've just upgraded, try removing your gls auxiliary
5473     files^^J and recompile}%
5474   \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`

```
5475 \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`

```
5476 \newcommand*{\glsgroupheading}[1]{}
```

It is possible to “trick” `makeindex` into treating entries as though they belong to the same group, even if the terms don’t start with the same letter, by modifying the sort key. For example, all entries belonging to one group could be defined so that the sort key starts with an a, while entries belonging to another group could be defined so that the sort key starts with a b, and so on. If you want each group to have a heading, you would then need to modify the translation control sequences `\glsgetgrouptitle` and `\glsgetgrouplabel` so that the label is translated into the required title (and vice-versa).

`\glsgetgrouptitle{<label>}`

This command produces the title for the glossary group whose label is given by  $\langle label \rangle$ . By default, the group labelled `glssymbols` produces `\glssymbolsgroupname`, the group labelled `glsnumbers` produces `\glsnumbersgroupname` and all the

other groups simply produce their label. As mentioned above, the group labels are: `glsymbols`, `glsnumbers`, A, ..., Z. If you want to redefine the group titles, you will need to redefine this command. Languages other than English may produce labels that are non-expandable, so we need to check for that otherwise it will create a “missing `\endcsname` inserted” error.

`\glsgetgrouptitle`

```
5477 \newcommand*{\glsgetgrouptitle}[1]{%
5478   \@gls@getgrouptitle{#1}{\@gls@grptitle}%
5479   \@gls@grptitle
5480 }
```

`\@gls@getgrouptitle` Gets the group title specified by the label (first argument) and stores in the second argument, which must be a control sequence.

```
5481 \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.

```
5482 \dtl@ifsingle{#1}%
5483 {%
5484   \ifcsundef{#1groupname}{\def#2{#1}}{\letcs#2{#1groupname}}%
5485 }%
5486 {%
5487   \ifboolexpr{test{\ifstrequal{#1}{glsymbols}}
5488               or test{\ifstrequal{#1}{glsnumbers}}}%
5489   {%
5490     \ifcsundef{#1groupname}{\def#2{#1}}{\letcs#2{#1groupname}}%
5491   }%
5492   {%
5493     \def#2{#1}%
5494   }%
5495 }%
5496 }
```

`@getothergrouptitle` Version for the no-indexing app option:

```
5497 \newcommand*{\@gls@noidx@getgrouptitle}[2]{%
5498   \DTLifint{#1}%
5499   {\edef#2{\char#1\relax}}%
5500   {%
5501     \ifcsundef{#1groupname}{\def#2{#1}}{\letcs#2{#1groupname}}%
5502   }%
5503 }
```

`\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`

```
5504 \newcommand*{\glsgetgrouplabel}[1]{%
5505 \ifthenelse{\equal{#1}{\glssymbolsgroupname}}{\glssymbols}{%
5506 \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`

```
5507 \newcommand*{\setentrycounter}[2][ ]{%
5508   \def\@glo@counterprefix{#1}%
5509   \ifx\@glo@counterprefix\@empty
5510     \def\@glo@counterprefix{.}%
5511   \else
5512     \def\@glo@counterprefix{.#1.}%
5513   \fi
5514   \def\glsentrycounter{#2}%
5515 }
```

The current glossary style can be set using `\setglossarystyle{<style>}`.

`\setglossarystyle`

```
5516 \newcommand*{\setglossarystyle}[1]{%
5517   \ifcsundef{@glsstyle@#1}%
5518   {%
5519     \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
5520   }%
5521   {%
5522     \csname @glsstyle@#1\endcsname
5523   }%
5524 }
```

`\glossarystyle`

```
5525 \newcommand*{\glossarystyle}[1]{%
5526   \ifcsundef{@glsstyle@#1}%
5527   {%
5528     \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
5529   }%
5530   {%
5531     \GlossariesWarning
5532     {Deprecated command \string\glossarystyle.^^J
5533     I recommend you switch to \string\setglossarystyle\space unless
5534     you want to maintain backward compatibility}%
5535     \setglossentrycompatibility
5536     \csname @glsstyle@#1\endcsname

5537   \ifcsdef{@glscompstyle@#1}%
5538   {\setglossentrycompatibility\csuse{@glscompstyle@#1}}%
5539   {}}%
```

```
5540 }%
5541 }
```

`\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.

```
5542 \newcommand{\newglossarystyle}[2]{%
5543   \ifcsundef{@glsstyle@#1}%
5544   {%
5545     \expandafter\def\csname @glsstyle@#1\endcsname{#2}%
5546   }%
5547   {%
5548     \PackageError{glossaries}{Glossary style ‘#1’ is already defined}{}%
5549   }%
5550 }
```

`\renewglossarystyle` Code for this macro supplied by Marco Daniel.

```
5551 \newcommand{\renewglossarystyle}[2]{%
5552   \ifcsundef{@glsstyle@#1}%
5553   {%
5554     \PackageError{glossaries}{Glossary style ‘#1’ isn’t already defined}{}%
5555   }%
5556   {%
5557     \csdef{@glsstyle@#1}{#2}%
5558   }%
5559 }
```

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`

```
5560 \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`

```
5561 \ifcsundef{hyperlink}%
5562 {%
5563   \def\glshypernumber#1{#1}%
5564 }%
5565 {%
5566   \def\glshypernumber#1{\@glshypernumber#1\nohyperpage{}}\@nil}
5567 }
```

`\@glshypernumber` This code was provided by Heiko Oberdiek to allow material to be attached to the location.

```
5568 \def\@glshypernumber#1\nohyperpage#2#3\@nil{%
5569   \ifx\#1\%
5570     \else
5571       \@delimR#1\delimR\delimR\%
5572     \fi
5573     \ifx\#2\%
5574       \else
5575         #2%
5576       \fi
5577       \ifx\#3\%
5578       \else
5579         \@glshypernumber#3\@nil
5580       \fi
5581 }
```

`\@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`

```
5582 \def\@delimR#1\delimR #2\delimR #3\{%
5583 \ifx\#2\%
5584   \@delimN{#1}%
5585 \else
5586   \@gls@numberlink{#1}\delimR\@gls@numberlink{#2}%
5587 \fi}
```

`\@delimN` displays a list of individual numbers, instead of a range:

\@delimN

```
5588 \def\@delimN#1{\@delimN#1\delimN \delimN\\}
5589 \def\@delimN#1\delimN #2\delimN#3\\{\%
5590 \ifx\\#3\\%
5591   \@gls@numberlink{#1}%
5592 \else
5593   \@gls@numberlink{#1}\delimN\@gls@numberlink{#2}%
5594 \fi
5595 }
```

The following code is modified from hyperref's \HyInd@pagelink where the name of the counter being used is given by \@gls@counter.

```
5596 \def\@gls@numberlink#1{%
5597 \begingroup
5598 \toks@={}%
5599 \@gls@removespaces#1 \@nil
5600 \endgroup}

5601 \def\@gls@removespaces#1 #2\@nil{%
5602 \toks@=\expandafter{\the\toks@#1}%
5603 \ifx\\#2\\%
5604   \edef\x{\the\toks@}%
5605   \ifx\x\empty
5606     \else

5607     \hyperlink{\glstrycounter\@glo@counterprefix\the\toks@}%
5608       {\the\toks@}%
5609   \fi
5610 \else
5611   \@gls@ReturnAfterFi{%
5612     \@gls@removespaces#2\@nil
5613   }%
5614 \fi
5615 }
5616 \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

```
5617 \newcommand*\hyperrm[1]{\textrm{\glshypernumber{#1}}}
```

\hypersf

```
5618 \newcommand*\hypersf[1]{\textsf{\glshypernumber{#1}}}
```

\hypertt

```
5619 \newcommand*\hypertt[1]{\texttt{\glshypernumber{#1}}}
```

```

\hyperbf
5620 \newcommand*{\hyperbf}[1]{\textbf{\glshypernumber{#1}}}

\hypermd
5621 \newcommand*{\hypermd}[1]{\textmd{\glshypernumber{#1}}}

\hyperit
5622 \newcommand*{\hyperit}[1]{\textit{\glshypernumber{#1}}}

\hypersl
5623 \newcommand*{\hypersl}[1]{\textsl{\glshypernumber{#1}}}

\hyperup
5624 \newcommand*{\hyperup}[1]{\textup{\glshypernumber{#1}}}

\hypersc
5625 \newcommand*{\hypersc}[1]{\textsc{\glshypernumber{#1}}}

\hyperemph
5626 \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.

```

5627 \newcommand{\oldacronym}[4][\gls@label]{%
5628   \def\gls@label{#2}%
5629   \newacronym[#4]{#1}{#2}{#3}%
5630   \ifcsundef{xspace}%
5631     {%

```



```

5632 \expandafter\edef\csname#1\endcsname{%
5633 \noexpand\@ifstar{\noexpand\Gls{#1}}{\noexpand\gls{#1}}}%
5634 }%
5635 }%
5636 {%
5637 \expandafter\edef\csname#1\endcsname{%
5638 \noexpand\@ifstar{\noexpand\Gls{#1}\noexpand\xspace}{%
5639 \noexpand\gls{#1}\noexpand\xspace}%
5640 }%
5641 }%
5642 }

```

`\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`

```

5643 \newcommand{\newacronym}[4] [] {}

```

Set up some convenient short cuts. These need to be changed if `\newacronym` is changed (or if the description key is changed).

`\acrpluralsuffix` Plural suffix used by `\newacronym`. This just defaults to `\glspluralsuffix` but is changed to include `\textup` if the `smallcaps` option is used, so that the suffix doesn't appear in small caps as it doesn't look right. For example, `ABCS` looks as though the “s” is part of the acronym, but `ABCs` looks as though the “s” is a plural suffix. Since the entire text `abcs` is set in `\textsc`, `\textup` is needed to cancel it out.

```

5644 \newcommand*{\acrpluralsuffix}{\glspluralsuffix}

```

If `garamondx` has been loaded, need to use `\textulc` instead of `\textup`.

`\glstextup`

```

5645 \newrobustcmd*{\glstextup}[1]{\ifdef\textulc{\textulc{#1}}{\textup{#1}}}

```

The following are defined for compatibility with version 2.07 and earlier.

`\glsshortkey`

```

5646 \newcommand*{\glsshortkey}{short}

```

```

\glsshortpluralkey
5647 \newcommand*{\glsshortpluralkey}{shortplural}

\glslongkey
5648 \newcommand*{\glslongkey}{long}

\glslongpluralkey
5649 \newcommand*{\glslongpluralkey}{longplural}

\acrfull  Full form of the acronym.
5650 \newrobustcmd*{\acrfull}{\@gls@hyp@opt\ns@acrfull}

5651 \newcommand*\ns@acrfull[2][\%
5652   \new@ifnextchar[\@acrfull{#1}{#2}]{\%
5653     {\@acrfull{#1}{#2}[]}\%
5654 }

\@acrfull  Low-level macro:
5655 \def\@acrfull#1#2[#3]{\%
    Make it easier for acronym styles to change this:
5656   \acrfullfmt{#1}{#2}{#3}\%
5657 }

    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.
5658 \newcommand*{\acrfullfmt}[3]{\%
5659   \acrlinkfullformat{\@acrlong}{\@acrshort}{#1}{#2}{#3}\%
5660 }

\acrlinkfullformat  Format for full links like \acrfull. Syntax: \acrlinkfullformat{<long
cs>}{<short cs>}{<options>}{<label>}{<insert>}
5661 \newcommand{\acrlinkfullformat}[5]{\%
5662   \acrfullformat{#1{#3}{#4}[#5]}{#2{#3}{#4}[]}\%
5663 }

\acrfullformat  Default full form is <long> (<short>).
5664 \newcommand{\acrfullformat}[2]{#1\glsspace(#2)}

\glsspace  Robust space to ensure it's written to the .glsdefs file.
5665 \newrobustcmd{\glsspace}{\space}

    Default format for full acronym

\Acrfull
5666 \newrobustcmd*{\Acrfull}{\@gls@hyp@opt\ns@Acrfull}

```

```

5667 \newcommand*\ns@Acrfull[2][]{%
5668   \new@ifnextchar[{\@Acrfull{#1}{#2}}{%
5669     {\@Acrfull{#1}{#2}}}%
5670 }

```

Low-level macro:

```

5671 \def\@Acrfull#1#2[#3]{%

```

Make it easier for acronym styles to change this:

```

5672   \Acrfullfmt{#1}{#2}{#3}%
5673 }

```

`\Acrfullfmt` First letter upper case full format.

```

5674 \newcommand*\@Acrfullfmt[3]{%
5675   \acrlinkfullformat{\@Acrlong}{\@acrshort}{#1}{#2}{#3}%
5676 }

```

`\ACRfull`

```

5677 \newrobustcmd*\@ACRfull{\@gls@hyp@opt\ns@ACRfull}

```

```

5678 \newcommand*\ns@ACRfull[2][]{%
5679   \new@ifnextchar[{\@ACRfull{#1}{#2}}{%
5680     {\@ACRfull{#1}{#2}}}%
5681 }

```

Low-level macro:

```

5682 \def\@ACRfull#1#2[#3]{%

```

Make it easier for acronym styles to change this:

```

5683   \ACRfullfmt{#1}{#2}{#3}%
5684 }

```

`\ACRfullfmt` All upper case full format.

```

5685 \newcommand*\@ACRfullfmt[3]{%
5686   \acrlinkfullformat{\@ACRlong}{\@ACRshort}{#1}{#2}{#3}%
5687 }

```

Plural:

`\acrfullpl`

```

5688 \newrobustcmd*\@acrfullpl{\@gls@hyp@opt\ns@acrfullpl}

```

```

5689 \newcommand*\ns@acrfullpl[2][]{%
5690   \new@ifnextchar[{\@acrfullpl{#1}{#2}}{%
5691     {\@acrfullpl{#1}{#2}}}%
5692 }

```

Low-level macro:

```

5693 \def\@acrfullpl#1#2[#3]{%

```

Make it easier for acronym styles to change this:

```
5694 \acrfullplfmt{#1}{#2}{#3}%  
5695 }
```

\acrfullplfmt No case change plural full format.

```
5696 \newcommand*\acrfullplfmt[3]{%  
5697 \acrlinkfullformat{\@acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%  
5698 }
```

\Acrfullpl

```
5699 \newrobustcmd*\Acrfullpl{\@gls@hyp@opt\ns@Acrfullpl}  
  
5700 \newcommand*\ns@Acrfullpl[2][{}]{%  
5701 \new@ifnextchar[{\@Acrfullpl{#1}{#2}}%  
5702 {\@Acrfullpl{#1}{#2}[]}%  
5703 }
```

Low-level macro:

```
5704 \def\@Acrfullpl#1#2[#3]{%
```

Make it easier for acronym styles to change this:

```
5705 \Acrfullplfmt{#1}{#2}{#3}%  
5706 }
```

\Acrfullplfmt First letter upper case plural full format.

```
5707 \newcommand*\Acrfullplfmt[3]{%  
5708 \acrlinkfullformat{\@Acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%  
5709 }
```

\ACRfullpl

```
5710 \newrobustcmd*\ACRfullpl{\@gls@hyp@opt\ns@ACRfullpl}  
  
5711 \newcommand*\ns@ACRfullpl[2][{}]{%  
5712 \new@ifnextchar[{\@ACRfullpl{#1}{#2}}%  
5713 {\@ACRfullpl{#1}{#2}[]}%  
5714 }
```

Low-level macro:

```
5715 \def\@ACRfullpl#1#2[#3]{%
```

Make it easier for acronym styles to change this:

```
5716 \ACRfullplfmt{#1}{#2}{#3}%  
5717 }
```

\ACRfullplfmt All upper case plural full format.

```
5718 \newcommand*\ACRfullplfmt[3]{%  
5719 \acrlinkfullformat{\@ACRlongpl}{\@ACRshortpl}{#1}{#2}{#3}%  
5720 }
```

## 1.17 Predefined acronym styles

`\acronymfont` This is only used with the additional acronym styles:

```
5721 \newcommand{\acronymfont}[1]{#1}
```

`\firstacronymfont` This is only used with the additional acronym styles:

```
5722 \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.

```
5723 \newcommand*{\acrnameformat}[2]{\acronymfont{#1}}
```

Define some tokens used by `\newacronym`:

`\glskeylisttok`

```
5724 \newtoks\glskeylisttok
```

`\glslabeltok`

```
5725 \newtoks\glslabeltok
```

`\glsshorttok`

```
5726 \newtoks\glsshorttok
```

`\glslongtok`

```
5727 \newtoks\glslongtok
```

`\newacronymhook` Provide a hook for `\newacronym`:

```
5728 \newcommand*{\newacronymhook}{}
```

`\setGenericNewAcronym` New improved version of setting the acronym style.

```
5729 \newcommand*{\SetGenericNewAcronym}{%
5730   \renewcommand{\newacronym}[4][[]]{%
5731     \ifdefempty{\@glsacronymlists}%
5732     {%
5733       \def\@glo@type{\acronymtype}%
5734       \setkeys{glossentry}{##1}%
5735       \DeclareAcronymList{\@glo@type}%
5736     }%
5737   }%
5738   \glskeylisttok{##1}%
5739   \glslabeltok{##2}%
5740   \glsshorttok{##3}%
5741   \glslongtok{##4}%
5742   \newacronymhook
5743   \protected@edef\@do@newglossaryentry{%
5744     \noexpand\newglossaryentry{\the\glslabeltok}%
5745     {%
5746       type=\acronymtype,%
```

```

5747     name={\expandonce{\acronymentry{##2}}},%
5748     sort={\acronymsort{\the\glsshorttok}{\the\glslongtok}},%
5749     text={\the\glsshorttok},%
5750     short={\the\glsshorttok},%
5751     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5752     long={\the\glslongtok},%
5753     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5754     \GenericAcronymFields,%
5755     \the\glskeylisttok
5756   }%
5757 }%
5758 \do@newglossaryentry
5759 }%

```

Make sure that \acrfull etc reflects the new style:

```

5760 \renewcommand*{\acrfullfmt}[3]{%
5761   \glslink[##1]{##2}{\genacrfullformat{##2}{##3}}}%
5762 \renewcommand*{\Acrfullfmt}[3]{%
5763   \glslink[##1]{##2}{\Genacrfullformat{##2}{##3}}}%
5764 \renewcommand*{\ACRfullfmt}[3]{%
5765   \glslink[##1]{##2}{%
5766     \mfirstucMakeUppercase{\genacrfullformat{##2}{##3}}}%
5767 \renewcommand*{\acrfullplfmt}[3]{%
5768   \glslink[##1]{##2}{\genplacrfullformat{##2}{##3}}}%
5769 \renewcommand*{\Acrfullplfmt}[3]{%
5770   \glslink[##1]{##2}{\Genplacrfullformat{##2}{##3}}}%
5771 \renewcommand*{\ACRfullplfmt}[3]{%
5772   \glslink[##1]{##2}{%
5773     \mfirstucMakeUppercase{\genplacrfullformat{##2}{##3}}}%

```

Make sure that \glsentryfull etc reflects the new style:

```

5774 \renewcommand*{\glsentryfull}[1]{\genacrfullformat{##1}{}}%
5775 \renewcommand*{\Glsentryfull}[1]{\Genacrfullformat{##1}{}}%
5776 \renewcommand*{\glsentryfullpl}[1]{\genplacrfullformat{##1}{}}%
5777 \renewcommand*{\Glsentryfullpl}[1]{\Genplacrfullformat{##1}{}}%
5778 }

```

`\GenericAcronymFields` Fields used by `\SetGenericNewAcronym` that can be changed by the acronym style.

```

5779 \newcommand*{\GenericAcronymFields}{description={\the\glslongtok}}

```

`\acronymentry` `\acronymentry{<label>}`

Display style for the name field in the list of acronyms.

```

5780 \newcommand*{\acronymentry}[1]{\acronymfont{\glsentryshort{#1}}}

```

`\acronymsort` `\acronymsort{<short>}{<long>}`

Default sort format for acronyms.

```
5781 \newcommand*{\acronymsort}[2]{#1}
```

`\setacronymstyle`    `\setacronymstyle{<style name>}`

```
5782 \newcommand*{\setacronymstyle}[1]{%
5783   \ifcsundef{@glsacr@dispstyle@#1}%
5784   {%
5785     \PackageError{glossaries}{Undefined acronym style ‘#1’}{}%
5786   }%
5787   {%
5788     \ifdefempty{\@glsacronymlists}%
5789     {%
5790       \DeclareAcronymList{<acronym type>%
5791     }%
5792     {}%
5793     \SetGenericNewAcronym
5794     \GlsUseAcrStyleDefs{#1}%
5795     \@for\@gls@type:=\@glsacronymlists\do{%
5796       \defglsentryfmt[\@gls@type]{\GlsUseAcrEntryDispStyle{#1}}%
5797     }%
5798   }%
5799 }
```

`\newacronymstyle`    `\newacronymstyle{<style name>}{<entry format definition>}{<display definitions>}`

Defines a new acronym style called <style name>.

```
5800 \newcommand*{\newacronymstyle}[3]{%
5801   \ifcsdef{@glsacr@dispstyle@#1}%
5802   {%
5803     \PackageError{glossaries}{Acronym style ‘#1’ already exists}{}%
5804   }%
5805   {%
5806     \csdef{@glsacr@dispstyle@#1}{#2}%
5807     \csdef{@glsacr@styledefs@#1}{#3}%
5808   }%
5809 }
```

`\renewacronymstyle`    Redefines the given acronym style.

```
5810 \newcommand*{\renewacronymstyle}[3]{%
5811   \ifcsdef{@glsacr@dispstyle@#1}%
5812   {%
5813     \csdef{@glsacr@dispstyle@#1}{#2}%
5814     \csdef{@glsacr@styledefs@#1}{#3}%
5815   }%
```

```

5816  {%
5817    \PackageError{glossaries}{Acronym style ‘#1’ doesn’t exist}{}%
5818  }%
5819 }

```

seAcrEntryDispStyle

```

5820 \newcommand*{\GlsUseAcrEntryDispStyle}[1]{\csuse{@glsacr@dispstyle@#1}}

```

\GlsUseAcrStyleDefs

```

5821 \newcommand*{\GlsUseAcrStyleDefs}[1]{\csuse{@glsacr@styledefs@#1}}

```

Predefined acronym styles:

long-short    *<long>* (*<short>*) acronym style.

```

5822 \newacronymstyle{long-short}%
5823 {%

```

Check for long form in case this is a mixed glossary.

```

5824   \ifglshaslong{\glslabel}{\glsgenacfmt}{\glsgenentryfmt}%
5825 }%
5826 {%
5827   \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
5828   \renewcommand*{\genacrfullformat}[2]{%
5829     \glsentrylong{##1}##2\space
5830     (\protect\firstacronymfont{\glsentryshort{##1}})%
5831   }%
5832   \renewcommand*{\Genacrfullformat}[2]{%
5833     \Glsentrylong{##1}##2\space
5834     (\protect\firstacronymfont{\glsentryshort{##1}})%
5835   }%
5836   \renewcommand*{\genplacrfullformat}[2]{%
5837     \glsentrylongpl{##1}##2\space
5838     (\protect\firstacronymfont{\glsentryshortpl{##1}})%
5839   }%
5840   \renewcommand*{\Genplacrfullformat}[2]{%
5841     \Glsentrylongpl{##1}##2\space
5842     (\protect\firstacronymfont{\glsentryshortpl{##1}})%
5843   }%
5844   \renewcommand*{\acronymentry}[1]{\acronymfont{\glsentryshort{##1}}}
5845   \renewcommand*{\acronymsort}[2]{##1}%
5846   \renewcommand*{\acronymfont}[1]{##1}%
5847   \renewcommand*{\firstacronymfont}[1]{\acronymfont{##1}}%
5848   \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
5849 }

```

short-long    *<short>* (*<long>*) acronym style.

```

5850 \newacronymstyle{short-long}%
5851 {%

```



Check for long form in case this is a mixed glossary.

```

5852 \ifglshaslong{\glslabel}{\glsgenacfmt}{\glsgenentryfmt}%
5853 }%
5854 {%
5855 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
5856 \renewcommand*{\genacrfullformat}[2]{%
5857 \protect\firstacronymfont{\glssentryshort{##1}}##2\space
5858 (\glssentrylong{##1})%
5859 }%
5860 \renewcommand*{\Genacrfullformat}[2]{%
5861 \protect\firstacronymfont{\Glsentryshort{##1}}##2\space
5862 (\glssentrylong{##1})%
5863 }%
5864 \renewcommand*{\genplacrfullformat}[2]{%
5865 \protect\firstacronymfont{\glssentryshortpl{##1}}##2\space
5866 (\glssentrylongpl{##1})%
5867 }%
5868 \renewcommand*{\Genplacrfullformat}[2]{%
5869 \protect\firstacronymfont{\Glsentryshortpl{##1}}##2\space
5870 (\glssentrylongpl{##1})%
5871 }%
5872 \renewcommand*{\acronymentry}[1]{\acronymfont{\glssentryshort{##1}}}%
5873 \renewcommand*{\acronymsort}[2]{##1}%
5874 \renewcommand*{\acronymfont}[1]{##1}%
5875 \renewcommand*{\firstacronymfont}[1]{\acronymfont{##1}}%
5876 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
5877 }

```

long-sc-short    *<long>* (\textsc{<short>}) acronym style.

```

5878 \newacronymstyle{long-sc-short}%
5879 {%
5880 \GlsUseAcrEntryDisplayStyle{long-short}%
5881 }%
5882 {%
5883 \GlsUseAcrStyleDefs{long-short}%
5884 \renewcommand{\acronymfont}[1]{\textsc{##1}}%
5885 \renewcommand*{\acrpluralsuffix}{\glstextup{\glspluralsuffix}}%
5886 }

```

long-sm-short    *<long>* (\textsmaller{<short>}) acronym style.

```

5887 \newacronymstyle{long-sm-short}%
5888 {%
5889 \GlsUseAcrEntryDisplayStyle{long-short}%
5890 }%
5891 {%
5892 \GlsUseAcrStyleDefs{long-short}%
5893 \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
5894 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
5895 }

```

sc-short-long    *<short>* (\textsc{<long>}) acronym style.

```

5896 \newacronymstyle{sc-short-long}%
5897 {%
5898   \GlsUseAcrEntryDisplayStyle{short-long}%
5899 }%
5900 {%
5901   \GlsUseAcrStyleDefs{short-long}%
5902   \renewcommand{\acronymfont}[1]{\textsc{##1}}%
5903   \renewcommand*{\acrpluralsuffix}{\glstextup{\glspluralsuffix}}%
5904 }
```

sm-short-long    *<short>* (\textsmaller{<long>}) acronym style.

```

5905 \newacronymstyle{sm-short-long}%
5906 {%
5907   \GlsUseAcrEntryDisplayStyle{short-long}%
5908 }%
5909 {%
5910   \GlsUseAcrStyleDefs{short-long}%
5911   \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
5912   \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
5913 }
```

long-short-desc    *<long>* ({<short>}) acronym style that has an accompanying description (which the user needs to supply).

```

5914 \newacronymstyle{long-short-desc}%
5915 {%
5916   \GlsUseAcrEntryDisplayStyle{long-short}%
5917 }%
5918 {%
5919   \GlsUseAcrStyleDefs{long-short}%
5920   \renewcommand*{\GenericAcronymFields}{}%
5921   \renewcommand*{\acronymsort}[2]{##2}%
5922   \renewcommand*{\acronymentry}[1]{%
5923     \glsentrylong{##1}\space (\acronymfont{\glsentryshort{##1}})}%
5924 }
```

long-sc-short-desc    *<long>* (\textsc{<short>}) acronym style that has an accompanying description (which the user needs to supply).

```

5925 \newacronymstyle{long-sc-short-desc}%
5926 {%
5927   \GlsUseAcrEntryDisplayStyle{long-sc-short}%
5928 }%
5929 {%
5930   \GlsUseAcrStyleDefs{long-sc-short}%
5931   \renewcommand*{\GenericAcronymFields}{}%
5932   \renewcommand*{\acronymsort}[2]{##2}%
5933   \renewcommand*{\acronymentry}[1]{%
5934     \glsentrylong{##1}\space (\textsc{\acronymfont{\glsentryshort{##1}})}%
5935 }
```

long-sm-short-desc    *<long>* (\textsmaller{<short>}) acronym style that has an accompanying description (which the user needs to supply).

```
5936 \newacronymstyle{long-sm-short-desc}%
5937 {%
5938   \GlsUseAcrEntryDispStyle{long-sm-short}%
5939 }%
5940 {%
5941   \GlsUseAcrStyleDefs{long-sm-short}%
5942   \renewcommand*{\GenericAcronymFields}{}%
5943   \renewcommand*{\acronymsort}[2]{##2}%
5944   \renewcommand*{\acronymentry}[1]{%
5945     \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
5946 }
```

short-long-desc    *<short>* ({<long>}) acronym style that has an accompanying description (which the user needs to supply).

```
5947 \newacronymstyle{short-long-desc}%
5948 {%
5949   \GlsUseAcrEntryDispStyle{short-long}%
5950 }%
5951 {%
5952   \GlsUseAcrStyleDefs{short-long}%
5953   \renewcommand*{\GenericAcronymFields}{}%
5954   \renewcommand*{\acronymsort}[2]{##2}%
5955   \renewcommand*{\acronymentry}[1]{%
5956     \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
5957 }
```

sc-short-long-desc    *<long>* (\textsc{<short>}) acronym style that has an accompanying description (which the user needs to supply).

```
5958 \newacronymstyle{sc-short-long-desc}%
5959 {%
5960   \GlsUseAcrEntryDispStyle{sc-short-long}%
5961 }%
5962 {%
5963   \GlsUseAcrStyleDefs{sc-short-long}%
5964   \renewcommand*{\GenericAcronymFields}{}%
5965   \renewcommand*{\acronymsort}[2]{##2}%
5966   \renewcommand*{\acronymentry}[1]{%
5967     \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
5968 }
```

sm-short-long-desc    *<long>* (\textsmaller{<short>}) acronym style that has an accompanying description (which the user needs to supply).

```
5969 \newacronymstyle{sm-short-long-desc}%
5970 {%
5971   \GlsUseAcrEntryDispStyle{sm-short-long}%
5972 }%
```

```

5973 {%
5974   \GlsUseAcrStyleDefs{sm-short-long}%
5975   \renewcommand*{\GenericAcronymFields}{}%
5976   \renewcommand*{\acronymsort}[2]{##2}%
5977   \renewcommand*{\acronymentry}[1]{%
5978     \glstrylong{##1}\space (\acronymfont{\glstryshort{##1}})}%
5979 }

```

dua *<long>* only acronym style.

```

5980 \newacronymstyle{dua}%
5981 {%

```

Check for long form in case this is a mixed glossary.

```

5982   \ifdefempty\glscustomtext
5983   {%
5984     \ifglshaslong{\glslabel}%
5985     {%
5986       \glssifplural
5987       {%

```

Plural form:

```

5988         \glscapscase
5989         {%

```

Plural form, don't adjust case:

```

5990         \glstrylongpl{\glslabel}\glssinsert
5991         }%
5992         {%

```

Plural form, make first letter upper case:

```

5993         \Glsentrylongpl{\glslabel}\glssinsert
5994         }%
5995         {%

```

Plural form, all caps:

```

5996         \mfirstucMakeUppercase
5997         {\glstrylongpl{\glslabel}\glssinsert}%
5998         }%
5999         }%
6000         {%

```

Singular form

```

6001         \glscapscase
6002         {%

```

Singular form, don't adjust case:

```

6003         \glstrylong{\glslabel}\glssinsert
6004         }%
6005         {%

```

Subsequent singular form, make first letter upper case:

```

6006         \Glsentrylong{\glslabel}\glssinsert

```

```

6007         }%
6008         {%

Subsequent singular form, all caps:
6009         \mfirstucMakeUppercase
6010         {\glsentrylong{\glslabel}\glsinsert}%
6011         }%
6012         }%
6013         }%
6014         {%

Not an acronym:
6015         \glsgenentryfmt
6016         }%
6017         }%
6018         {\glscustomtext\glsinsert}%
6019 }%
6020 {%
6021 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%

6022 \renewcommand*{\acrfullfmt}[3]{%
6023     \glslink[##1]{##2}{\glsentrylong{##2}##3\space
6024     (\acronymfont{\glsentryshort{##2}})}}%
6025 \renewcommand*{\Acrfullfmt}[3]{%
6026     \glslink[##1]{##2}{\Glsentrylong{##2}##3\space
6027     (\acronymfont{\glsentryshort{##2}})}}%
6028 \renewcommand*{\ACRfullfmt}[3]{%
6029     \glslink[##1]{##2}{%
6030     \mfirstucMakeUppercase{\glsentrylong{##2}##3\space
6031     (\acronymfont{\glsentryshort{##2}})}}}%

6032 \renewcommand*{\acrfullplfmt}[3]{%
6033     \glslink[##1]{##2}{\glsentrylongpl{##2}##3\space
6034     (\acronymfont{\glsentryshortpl{##2}})}}%

6035 \renewcommand*{\Acrfullplfmt}[3]{%
6036     \glslink[##1]{##2}{\Glsentrylongpl{##2}##3\space
6037     (\acronymfont{\glsentryshortpl{##2}})}}%
6038 \renewcommand*{\ACRfullplfmt}[3]{%
6039     \glslink[##1]{##2}{%
6040     \mfirstucMakeUppercase{\glsentrylongpl{##2}##3\space
6041     (\acronymfont{\glsentryshortpl{##2}})}}}%
6042 \renewcommand*{\glsentryfull}[1]{%
6043     \glsentrylong{##1}\space(\acronymfont{\glsentryshort{##1}})%
6044 }%
6045 \renewcommand*{\Glsentryfull}[1]{%
6046     \Glsentrylong{##1}\space(\acronymfont{\glsentryshort{##1}})%
6047 }%
6048 \renewcommand*{\glsentryfullpl}[1]{%
6049     \glsentrylongpl{##1}\space(\acronymfont{\glsentryshortpl{##1}})%
6050 }%

```

```

6051 \renewcommand*\Glsentryfullpl}[1]{%
6052   \Glsentrylongpl{##1}\space(\acronymfont{\glentryshortpl{##1}})%
6053 }%
6054 \renewcommand*\acronymentry}[1]{\acronymfont{\glentryshort{##1}}}%
6055 \renewcommand*\acronymsort}[2]{##1}%
6056 \renewcommand*\acronymfont}[1]{##1}%
6057 \renewcommand*\acrpluralsuffix{\glspluralsuffix}%
6058 }

```

dua-desc *<long>* only acronym style with user-supplied description.

```

6059 \newacronymstyle{dua-desc}%
6060 {%
6061   \GlsUseAcrEntryDispStyle{dua}%
6062 }%
6063 {%
6064   \GlsUseAcrStyleDefs{dua}%
6065   \renewcommand*\GenericAcronymFields{}%
6066   \renewcommand*\acronymentry}[1]{\acronymfont{\glentrylong{##1}}}%
6067   \renewcommand*\acronymsort}[2]{##2}%
6068 }%

```

footnote *<short>*\footnote{*<long>*} acronym style.

```

6069 \newacronymstyle{footnote}%
6070 {%

```

Check for long form in case this is a mixed glossary.

```

6071 \ifglshaslong{\glslabel}{\glsgenacfmt}{\glsgenentryfmt}%
6072 }%
6073 {%
6074   \renewcommand*\GenericAcronymFields{description={\the\glslongtok}}%

```

Need to ensure hyperlinks are switched off on first use:

```

6075 \glshyperfirstfalse
6076 \renewcommand*\genacrfullformat}[2]{%
6077   \protect\firstacronymfont{\glentryshort{##1}}##2%
6078   \protect\footnote{\glentrylong{##1}}%
6079 }%
6080 \renewcommand*\Genacrfullformat}[2]{%
6081   \firstacronymfont{\glentryshort{##1}}##2%
6082   \protect\footnote{\glentrylong{##1}}%
6083 }%
6084 \renewcommand*\genplacrfullformat}[2]{%
6085   \protect\firstacronymfont{\glentryshortpl{##1}}##2%
6086   \protect\footnote{\glentrylongpl{##1}}%
6087 }%
6088 \renewcommand*\Genplacrfullformat}[2]{%
6089   \protect\firstacronymfont{\glentryshortpl{##1}}##2%
6090   \protect\footnote{\glentrylongpl{##1}}%
6091 }%

```

```

6092 \renewcommand*{\acronymentry}[1]{\acronymfont{\glentryshort{##1}}}%
6093 \renewcommand*{\acronymsort}[2]{##1}%
6094 \renewcommand*{\acronymfont}[1]{##1}%
6095 \renewcommand*{\acrpluralsuffix}{\glpluralsuffix}%

```

Don't use footnotes for \acrfull:

```

6096 \renewcommand*{\acrfullfmt}[3]{%
6097   \glslink[##1]{##2}{\acronymfont{\glentryshort{##2}}##3\space
6098   (\glentrylong{##2})}%
6099 \renewcommand*{\Acrfullfmt}[3]{%
6100   \glslink[##1]{##2}{\acronymfont{\Glentryshort{##2}}##3\space
6101   (\glentrylong{##2})}%
6102 \renewcommand*{\ACRfullfmt}[3]{%
6103   \glslink[##1]{##2}{%
6104     \mfirstucMakeUppercase{\acronymfont{\glentryshort{##2}}##3\space
6105     (\glentrylong{##2})}}}%
6106 \renewcommand*{\acrfullplfmt}[3]{%
6107   \glslink[##1]{##2}{\acronymfont{\glentryshortpl{##2}}##3\space
6108   (\glentrylongpl{##2})}%
6109 \renewcommand*{\Acrfullplfmt}[3]{%
6110   \glslink[##1]{##2}{\acronymfont{\Glentryshortpl{##2}}##3\space
6111   (\glentrylongpl{##2})}%
6112 \renewcommand*{\ACRfullplfmt}[3]{%
6113   \glslink[##1]{##2}{%
6114     \mfirstucMakeUppercase{\acronymfont{\glentryshortpl{##2}}##3\space
6115     (\glentrylongpl{##2})}}}%

```

Similarly for \glentryfull etc:

```

6116 \renewcommand*{\glentryfull}[1]{%
6117   \acronymfont{\glentryshort{##1}}\space(\glentrylong{##1})}%
6118 \renewcommand*{\Glentryfull}[1]{%
6119   \acronymfont{\Glentryshort{##1}}\space(\glentrylong{##1})}%
6120 \renewcommand*{\glentryfullpl}[1]{%
6121   \acronymfont{\glentryshortpl{##1}}\space(\glentrylongpl{##1})}%
6122 \renewcommand*{\Glentryfullpl}[1]{%
6123   \acronymfont{\Glentryshortpl{##1}}\space(\glentrylongpl{##1})}%
6124 }

```

footnote-sc \textsc{<short>}\footnote{<long>} acronym style.

```

6125 \newacronymstyle{footnote-sc}%
6126 {%
6127   \GlsUseAcrEntryDispStyle{footnote}%
6128 }%
6129 {%
6130   \GlsUseAcrStyleDefs{footnote}%
6131   \renewcommand{\acronymentry}[1]{\acronymfont{\glentryshort{##1}}}
6132   \renewcommand{\acronymfont}[1]{\textsc{##1}}%
6133   \renewcommand*{\acrpluralsuffix}{\glstextup{\glpluralsuffix}}%
6134 }%

```

footnote-sm \textsmaller{<short>}\footnote{<long>} acronym style.

```
6135 \newacronymstyle{footnote-sm}%
6136 {%
6137   \GlsUseAcrEntryDispStyle{footnote}%
6138 }%
6139 {%
6140   \GlsUseAcrStyleDefs{footnote}%
6141   \renewcommand{\acronymentry}[1]{\acronymfont{\glentryshort{##1}}}
6142   \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
6143   \renewcommand*{\acrpluralsuffix}{\glpluralsuffix}%
6144 }%
```

footnote-desc <short>\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```
6145 \newacronymstyle{footnote-desc}%
6146 {%
6147   \GlsUseAcrEntryDispStyle{footnote}%
6148 }%
6149 {%
6150   \GlsUseAcrStyleDefs{footnote}%
6151   \renewcommand*{\GenericAcronymFields}{}%
6152   \renewcommand*{\acronymsort}[2]{##2}%
6153   \renewcommand*{\acronymentry}[1]{%
6154     \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
6155 }
```

footnote-sc-desc \textsc{<short>}\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```
6156 \newacronymstyle{footnote-sc-desc}%
6157 {%
6158   \GlsUseAcrEntryDispStyle{footnote-sc}%
6159 }%
6160 {%
6161   \GlsUseAcrStyleDefs{footnote-sc}%
6162   \renewcommand*{\GenericAcronymFields}{}%
6163   \renewcommand*{\acronymsort}[2]{##2}%
6164   \renewcommand*{\acronymentry}[1]{%
6165     \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
6166 }
```

footnote-sm-desc \textsmaller{<short>}\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```
6167 \newacronymstyle{footnote-sm-desc}%
6168 {%
6169   \GlsUseAcrEntryDispStyle{footnote-sm}%
6170 }%
6171 {%
6172   \GlsUseAcrStyleDefs{footnote-sm}%

```



```

6173 \renewcommand*{\GenericAcronymFields}{}%
6174 \renewcommand*{\acronymsort}[2]{##2}%
6175 \renewcommand*{\acronymentry}[1]{%
6176     \glsentrylong{##1}\space (\acronymfont{\glsentryshort{##1}})}%
6177 }

```

`\fineAcronymSynonyms`

```

6178 \newcommand*{\DefineAcronymSynonyms}{%

```

Short form

`\acs`

```

6179 \let\acs\acrshort

```

First letter uppercase short form

`\Acs`

```

6180 \let\Acs\Acrshort

```

Plural short form

`\acsp`

```

6181 \let\acsp\acrshortpl

```

First letter uppercase plural short form

`\Acsp`

```

6182 \let\Acsp\Acrshortpl

```

Long form

`\acl`

```

6183 \let\acl\aclong

```

Plural long form

`\aclp`

```

6184 \let\aclp\aclongpl

```

First letter upper case long form

`\Acl`

```

6185 \let\Acl\Aclong

```

First letter upper case plural long form

`\Aclp`

```

6186 \let\Aclp\Aclongpl

```

Full form

`\acf`

```

6187 \let\acf\acrfull

```

Plural full form

`\acfp`

```
6188 \let\acfp\acrfullpl
```

First letter upper case full form

`\Acf`

```
6189 \let\Acf\Acrfull
```

First letter upper case plural full form

`\Acfp`

```
6190 \let\Acfp\Acrfullpl
```

Standard form

`\ac`

```
6191 \let\ac\gls
```

First upper case standard form

`\Ac`

```
6192 \let\Ac\Gls
```

Standard plural form

`\acp`

```
6193 \let\acp\glspl
```

Standard first letter upper case plural form

`\Acp`

```
6194 \let\Acp\Glspl
```

```
6195 }
```

Define synonyms if required

```
6196 \ifglsacrshortcuts
```

```
6197 \DefineAcronymSynonyms
```

```
6198 \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.

```
6199 \newcommand*\SetDefaultAcronymDisplayStyle}[1]{%
```

```
6200 \defglsentryfmt[#1]{\glsentryfmt}%
```

```
6201 }
```

`\defaultNewAcronymDef` Sets up the acronym definition for the default style. The information is provided by the tokens `\glslabeltok`, `\glsshorttok`, `\glslongtok` and `\glskeylisttok`.

```

6202 \newcommand*{\DefaultNewAcronymDef}{%
6203   \edef\@do@newglossaryentry{%
6204     \noexpand\newglossaryentry{\the\glslabeltok}%
6205     {%
6206       type=\acronymtype,%
6207       name={\the\glsshorttok},%
6208       sort={\the\glsshorttok},%
6209       text={\the\glsshorttok},%
6210       first={\acrfullformat{\the\glslongtok}{\the\glsshorttok}},%
6211       plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6212       firstplural={\acrfullformat{\noexpand\expandonce\noexpand\@glo@longpl}%
6213                    {\noexpand\expandonce\noexpand\@glo@shortpl}},%
6214       short={\the\glsshorttok},%
6215       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6216       long={\the\glslongtok},%
6217       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6218       description={\the\glslongtok},%
6219       descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%

```

Remaining options specified by the user:

```

6220   \the\glskeylisttok
6221   }%
6222 }%
6223 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6224 \let\@org@gls@assign@plural\gls@assign@plural
6225 \let\@org@gls@assign@descplural\gls@assign@descplural
6226 \def\gls@assign@firstpl##1##2{%
6227   \@@gls@expand@field{##1}{firstpl}{##2}%
6228 }%
6229 \def\gls@assign@plural##1##2{%
6230   \@@gls@expand@field{##1}{plural}{##2}%
6231 }%
6232 \def\gls@assign@descplural##1##2{%
6233   \@@gls@expand@field{##1}{descplural}{##2}%
6234 }%
6235 \@do@newglossaryentry
6236 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6237 \let\gls@assign@plural\@org@gls@assign@plural
6238 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6239 }

```

`\DefaultAcronymStyle` Set up the default acronym style:

```

6240 \newcommand*{\SetDefaultAcronymStyle}{%
  Set the display style:
6241   \@for\@gls@type:=\@glsacronymlists\do{%
6242     \SetDefaultAcronymDisplayStyle{\@gls@type}%

```

6243 }%

Set up the definition of \newacronym:

6244 \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).

```
6245 \ifx\@glsacronymlists\@empty
6246 \def\@glo@type{\acronymtype}%
6247 \setkeys{glossentry}{##1}%
6248 \DeclareAcronymList{\@glo@type}%
6249 \SetDefaultAcronymDisplayStyle{\@glo@type}%
6250 \fi
6251 \glskeylisttok{##1}%
6252 \glslabeltok{##2}%
6253 \glsshorttok{##3}%
6254 \gslongtok{##4}%
6255 \newacronymhook
6256 \DefaultNewAcronymDef
6257 }%
6258 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
6259 }
```

\acrfootnote Used by the footnote acronym styles.

6260 \newcommand\*{\acrfootnote}[3]{\acrlinkfootnote{#1}{#2}{#3}}

\acrlinkfootnote

```
6261 \newcommand*{\acrlinkfootnote}[3]{%
6262 \footnote{\glslink[#1]{#2}{#3}}%
6263 }
```

\acrno linkfootnote

```
6264 \newcommand*{\acrno linkfootnote}[3]{%
6265 \footnote{#3}%
6266 }
```

AcronymDisplayStyle Sets the acronym display style for given glossary for the description and footnote combination.

```
6267 \newcommand*{\SetDescriptionFootnoteAcronymDisplayStyle}[1]{%
6268 \def\glsentryfmt[#1]{%

6269 \ifdefempty\glscustomtext
6270 {%
6271 \ifglsused{\glslabel}%
6272 {%
6273 \acronymfont{\glsentryfmt}%
6274 }%
6275 }
```

```

6276 \firstacronymfont{\glsgenentryfmt}%
6277 \ifglshassymbol{\glslabel}%
6278 {%
6279 \expandafter\protect\expandafter\acrfootnote\expandafter
6280 {\@gls@link@opts}{\@gls@link@label}%
6281 {%
6282 \glsifplural
6283 {\glsentrysymbolplural{\glslabel}}%
6284 {\glsentrysymbol{\glslabel}}%
6285 }%
6286 }%
6287 }%
6288 }%
6289 {\glscustomtext\glsinsert}%
6290 }%
6291 }

```

otnoteNewAcronymDef

```

6292 \newcommand*{\DescriptionFootnoteNewAcronymDef}{%
6293 \edef\@do@newglossaryentry{%
6294 \noexpand\newglossaryentry{\the\glslabeltok}%
6295 {%
6296 type=\acronymtype,%
6297 name={\noexpand\acronymfont{\the\glsshorttok}},%
6298 sort={\the\glsshorttok},%
6299 first={\the\glsshorttok},%
6300 firstplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6301 text={\the\glsshorttok},%
6302 plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6303 short={\the\glsshorttok},%
6304 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6305 long={\the\glslongtok},%
6306 longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6307 symbol={\the\glslongtok},%
6308 symbolplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6309 \the\glskeylisttok
6310 }%
6311 }%
6312 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6313 \let\@org@gls@assign@plural\gls@assign@plural
6314 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6315 \def\gls@assign@firstpl##1##2{%
6316 \@@gls@expand@field{##1}{firstpl}{##2}%
6317 }%
6318 \def\gls@assign@plural##1##2{%
6319 \@@gls@expand@field{##1}{plural}{##2}%
6320 }%
6321 \def\gls@assign@symbolplural##1##2{%
6322 \@@gls@expand@field{##1}{symbolplural}{##2}%

```

```

6323 }%
6324 \@do@newglossaryentry
6325 \let\gls@assign@plural\@org@gls@assign@plural
6326 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6327 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6328 }

```

`ootnoteAcronymStyle` If a description and footnote are both required, store the long form in the symbol key. Store the short form in text key. Note that since the long form is stored in the symbol key, if you want the long form to appear in the list of acronyms, you need to use a glossary style that displays the symbol key.

```

6329 \newcommand*\SetDescriptionFootnoteAcronymStyle{%
6330   \renewcommand{\newacronym}[4][\]{%
6331     \ifx\@glsacronymlists\@empty
6332       \def\@glo@type{\acronymtype}%
6333       \setkeys{glossentry}{##1}%
6334       \DeclareAcronymList{\@glo@type}%
6335       \SetDescriptionFootnoteAcronymDisplayStyle{\@glo@type}%
6336     \fi
6337     \glskeylisttok{##1}%
6338     \glslabeltok{##2}%
6339     \glsshorttok{##3}%
6340     \gslongtok{##4}%
6341     \newacronymhook
6342     \DescriptionFootnoteNewAcronymDef
6343   }%

```

If footnote package option is specified, set the first use to append the long form (stored in symbol) as a footnote.

```

6344 \@for\@gls@type:=\@glsacronymlists\do{%
6345   \SetDescriptionFootnoteAcronymDisplayStyle{\@gls@type}%
6346 }%

```

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.

```

6347 \ifglsacrsmallcaps
6348   \renewcommand*\acronymfont{[1]{\textsc{##1}}}%
6349   \renewcommand*\acrpluralsuffix{%
6350     \glstextup{\glspluralsuffix}}%
6351 \else
6352   \ifglsacrsmaller
6353     \renewcommand*\acronymfont{[1]{\textsmaller{##1}}}%
6354   \fi
6355 \fi

```

Check for package option clash

```

6356 \ifglsacrdua
6357   \PackageError{glossaries}{Option clash: 'footnote' and 'dua'}

```

```

6358     can't both be set}{}%
6359   \fi
6360 }%

```

**AcronymDisplayStyle** Sets the acronym display style for given glossary with description and dua combination.

```

6361 \newcommand*{\SetDescriptionDUAAcronymDisplayStyle}[1]{%
6362   \def\glentryfmt[#1]{\glsgenentryfmt}%
6363 }

```

**ionDUANewAcronymDef**

```

6364 \newcommand*{\DescriptionDUANewAcronymDef}{%
6365   \edef\@do@newglossaryentry{%
6366     \noexpand\newglossaryentry{\the\glslabeltok}%
6367     {%
6368       type=\acronymtype,%
6369       name={\the\glslongtok},%
6370       sort={\the\glslongtok},%
6371       text={\the\glslongtok},%
6372       first={\the\glslongtok},%
6373       plural={\noexpand\expandonce\noexpand\@glo@longpl},%
6374       firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6375       short={\the\glsshorttok},%
6376       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6377       long={\the\glslongtok},%
6378       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6379       symbol={\the\glsshorttok},%
6380       symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6381       \the\glskeylisttok
6382     }%
6383   }%
6384   \let\@org@gls@assign@firstpl\gls@assign@firstpl
6385   \let\@org@gls@assign@plural\gls@assign@plural
6386   \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6387   \def\gls@assign@firstpl##1##2{%
6388     \@@gls@expand@field{##1}{firstpl}{##2}%
6389   }%
6390   \def\gls@assign@plural##1##2{%
6391     \@@gls@expand@field{##1}{plural}{##2}%
6392   }%
6393   \def\gls@assign@symbolplural##1##2{%
6394     \@@gls@expand@field{##1}{symbolplural}{##2}%
6395   }%
6396   \@do@newglossaryentry
6397   \let\gls@assign@firstpl\@org@gls@assign@firstpl
6398   \let\gls@assign@plural\@org@gls@assign@plural
6399   \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6400 }

```

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.

```

6401 \newcommand*\SetDescriptionDUAAcronymStyle{%
6402   \ifglsmallcaps
6403     \PackageError{glossaries}{Option clash: 'smallcaps' and 'dua'
6404       can't both be set}{}%
6405   \else
6406     \ifglsmaller
6407       \PackageError{glossaries}{Option clash: 'smaller' and 'dua'
6408         can't both be set}{}%
6409     \fi
6410   \fi
6411   \renewcommand{\newacronym}[4][[]]{%
6412     \ifx\@glsacronymlists\@empty
6413       \def\@glo@type{\acronymtype}%
6414       \setkeys{glossentry}{##1}%
6415       \DeclareAcronymList{\@glo@type}%
6416       \SetDescriptionDUAAcronymDisplayStyle{\@glo@type}%
6417     \fi
6418     \glskeylisttok{##1}%
6419     \glslabeltok{##2}%
6420     \glsshorttok{##3}%
6421     \glslongtok{##4}%
6422     \newacronymhook
6423     \DescriptionDUANewAcronymDef
6424   }%

Set display.
6425   \@for\@gls@type:=\@glsacronymlists\do{%
6426     \SetDescriptionDUAAcronymDisplayStyle{\@gls@type}%
6427   }%
6428 }%
```

AcronymDisplayStyle Sets the acronym display style for given glossary using the description setting (but not footnote or dua).

```

6429 \newcommand*\SetDescriptionAcronymDisplayStyle}[1]{%
6430   \defglentryfmt[#1]{%

6431     \ifdefempty\glscustomtext
6432     {%
6433       \ifglused{\glslabel}%
6434       {%

Move the inserted text outside of \acronymfont
6435         \let\gls@org@insert\glsinsert
6436         \let\glsinsert\@empty
6437         \acronymfont{\glsgenentryfmt}\gls@org@insert
6438       }%

```



```

6439      {%
6440        \glsgenentryfmt
6441        \ifglshassymbol{\glslabel}%
6442        {%
6443          \glsifplural
6444          {%
6445            \def\@glo@symbol{\glsentrysymbolplural{\glslabel}}%
6446          }%
6447          {%
6448            \def\@glo@symbol{\glsentrysymbol{\glslabel}}%
6449          }%
6450          \space(\protect\firstacronymfont
6451          {\glscapscase
6452           {\@glo@symbol}
6453           {\@glo@symbol}
6454           {\mfirstucMakeUppercase{\@glo@symbol}}})%
6455        }%
6456      }%
6457    }%
6458  }%
6459  {\glscustomtext\glsinsert}%
6460 }%
6461 }

```

ptionNewAcronymDef

```

6462 \newcommand*{\DescriptionNewAcronymDef}{%
6463   \edef\@do@newglossaryentry{%
6464     \noexpand\newglossaryentry{\the\glslabeltok}%
6465     {%
6466       type=\acronymtype,%
6467       name={\noexpand
6468         \acrnameformat{\the\glsshorttok}{\the\glslongtok}},%
6469       sort={\the\glsshorttok},%
6470       first={\the\glslongtok},%
6471       firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6472       text={\the\glsshorttok},%
6473       plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6474       short={\the\glsshorttok},%
6475       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6476       long={\the\glslongtok},%
6477       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6478       symbol={\noexpand\@glo@text},%
6479       symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6480       \the\glskeylisttok}%
6481   }%
6482   \let\@org@gls@assign@firstpl\gls@assign@firstpl
6483   \let\@org@gls@assign@plural\gls@assign@plural
6484   \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6485   \def\gls@assign@firstpl##1##2{%

```

```

6486 \@@gls@expand@field{##1}{firstpl}{##2}%
6487 }%
6488 \def\gls@assign@plural##1##2{%
6489 \@@gls@expand@field{##1}{plural}{##2}%
6490 }%
6491 \def\gls@assign@symbolplural##1##2{%
6492 \@@gls@expand@field{##1}{symbolplural}{##2}%
6493 }%
6494 \do@newglossaryentry
6495 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6496 \let\gls@assign@plural\@org@gls@assign@plural
6497 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6498 }

```

**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.

```

6499 \newcommand*{\SetDescriptionAcronymStyle}{%
6500 \renewcommand{\newacronym}[4][\]{%
6501 \ifx\@glsacronymlists\@empty
6502 \def\@glo@type{\acronymtype}%
6503 \setkeys{glossentry}{##1}%
6504 \DeclareAcronymList{\@glo@type}%
6505 \SetDescriptionAcronymDisplayStyle{\@glo@type}%
6506 \fi
6507 \glskeylisttok{##1}%
6508 \glslabeltok{##2}%
6509 \glsshorttok{##3}%
6510 \glslongtok{##4}%
6511 \newacronymhook
6512 \DescriptionNewAcronymDef
6513 }%

```

Set display.

```

6514 \@for\@gls@type:=\@glsacronymlists\do{%
6515 \SetDescriptionAcronymDisplayStyle{\@gls@type}%
6516 }%

```

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.

```

6517 \ifglsacrsmallcaps
6518 \renewcommand{\acronymfont}[1]{\textsc{##1}}
6519 \renewcommand*{\acrpluralsuffix}{%
6520 \glstextup{\glspluralsuffix}}%
6521 \else
6522 \ifglsacrsmaller
6523 \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%

```

```

6524 \fi
6525 \fi
6526 }%

```

**AcronymDisplayStyle** Sets the acronym display style for given glossary with footnote setting (but not description or dua).

```

6527 \newcommand*{\SetFootnoteAcronymDisplayStyle}[1]{%
6528 \def\glsentryfmt[#1]{%

```

```

6529 \ifdefempty\glscustomtext
6530 {%

```

Move the inserted text outside of \acronymfont

```

6531 \let\gls@org@insert\glsinsert
6532 \let\glsinsert\@empty
6533 \ifglssused{\glslabel}%
6534 {%
6535 \acronymfont{\glsentryfmt}\gls@org@insert
6536 }%
6537 {%
6538 \firstacronymfont{\glsentryfmt}\gls@org@insert
6539 \ifglshaslong{\glslabel}%
6540 {%
6541 \expandafter\protect\expandafter\acrfootnote\expandafter
6542 {\@gls@link@opts}{\@gls@link@label}%
6543 {%
6544 \glsifplural
6545 {\glsentrylongpl{\glslabel}}%
6546 {\glsentrylong{\glslabel}}%
6547 }%
6548 }%
6549 }%
6550 }%
6551 }%
6552 {\glscustomtext\glsinsert}%
6553 }%
6554 }

```

**FootnoteNewAcronymDef**

```

6555 \newcommand*{\FootnoteNewAcronymDef}{%
6556 \edef\@do@newglossaryentry{%
6557 \noexpand\newglossaryentry{\the\glslabeltok}%
6558 {%
6559 type=\acronymtype,%
6560 name={\noexpand\acronymfont{\the\glsshorttok}},%
6561 sort={\the\glsshorttok},%
6562 text={\the\glsshorttok},%
6563 plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6564 first={\the\glsshorttok},%

```

```

6565     firstplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6566     short={\the\glsshorttok},%
6567     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6568     long={\the\glslongtok},%
6569     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6570     description={\the\glslongtok},%
6571     descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6572     \the\glskeylisttok
6573 }%
6574 }%
6575 \let\@org@gls@assign@plural\gls@assign@plural
6576 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6577 \let\@org@gls@assign@descplural\gls@assign@descplural
6578 \def\gls@assign@firstpl##1##2{%
6579   \@@gls@expand@field{##1}{firstpl}{##2}%
6580 }%
6581 \def\gls@assign@plural##1##2{%
6582   \@@gls@expand@field{##1}{plural}{##2}%
6583 }%
6584 \def\gls@assign@descplural##1##2{%
6585   \@@gls@expand@field{##1}{descplural}{##2}%
6586 }%
6587 \do@newglossaryentry
6588 \let\gls@assign@plural\@org@gls@assign@plural
6589 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6590 \let\gls@assign@descplural\@org@gls@assign@descplural
6591 }

```

**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.

```

6592 \newcommand*\SetFootnoteAcronymStyle{%
6593   \renewcommand{\newacronym}[4][\]{%
6594     \ifx\@glsacronymlists\@empty
6595       \def\@glo@type{\acronymtype}%
6596       \setkeys{glossentry}{##1}%
6597       \DeclareAcronymList{\@glo@type}%
6598       \SetFootnoteAcronymDisplayStyle{\@glo@type}%
6599     \fi
6600     \glskeylisttok{##1}%
6601     \glslabeltok{##2}%
6602     \glsshorttok{##3}%
6603     \glslongtok{##4}%
6604     \newacronymhook
6605     \FootnoteNewAcronymDef
6606   }%

```

Set display

```

6607   \@for\@gls@type:=\@glsacronymlists\do{%

```

```

6608 \SetFootnoteAcronymDisplayStyle{\@gls@type}%
6609 }%

```

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.

```

6610 \ifglsacrsmallcaps
6611 \renewcommand*{\acronymfont}[1]{\textsc{##1}}%
6612 \renewcommand*{\acrpluralsuffix}{%
6613 \glstextup{\glspluralsuffix}}%
6614 \else
6615 \ifglsacrsmaller
6616 \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%
6617 \fi
6618 \fi

```

Check for option clash

```

6619 \ifglsacrdua
6620 \PackageError{glossaries}{Option clash: ‘footnote’ and ‘dua’
6621 can’t both be set}}{}%
6622 \fi
6623 }%

```

`\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.

```

6624 \DeclareRobustCommand*{\glsdoparenifnotempty}[2]{%
6625 \protected@edef\gls@tmp{#1}%
6626 \ifdefempty\gls@tmp
6627 {}%
6628 {%
6629 \ifx\gls@tmp\@gls@default@value
6630 \else
6631 \space (#2{#1})%
6632 \fi
6633 }%
6634 }

```

`\AcronymDisplayStyle` Sets the acronym display style for given glossary where neither footnote nor description is required, but smallcaps or smaller specified.

```

6635 \newcommand*{\SetSmallAcronymDisplayStyle}[1]{%
6636 \defglsentryfmt[#1]{%
6637 \ifdefempty\glscustomtext
6638 {%

```

Move the inserted text outside of `\acronymfont`

```

6639 \let\gls@org@insert\glsinsert
6640 \let\glsinsert\@empty
6641 \ifglsused{\glslabel}%

```

```

6642      {%
6643      \acronymfont{\glsgenentryfmt}\gls@org@insert
6644      }%
6645      {%
6646      \glsgenentryfmt
6647      \ifglshassymbol{\glslabel}%
6648      {%
6649      \glsifplural
6650      {%
6651      \def\@glo@symbol{\glsentrysymbolplural{\glslabel}}%
6652      }%
6653      {%
6654      \def\@glo@symbol{\glsentrysymbol{\glslabel}}%
6655      }%
6656      \space
6657      (\glscapscase
6658      {\firstacronymfont{\@glo@symbol}}%
6659      {\firstacronymfont{\@glo@symbol}}%
6660      {\firstacronymfont{\mfirstucMakeUppercase{\@glo@symbol}}})%
6661      }%
6662      {}%
6663      }%
6664      }%
6665      {\glscustomtext\glsinsert}%
6666      }%
6667      }

```

\SmallNewAcronymDef

```

6668 \newcommand*{\SmallNewAcronymDef}{%
6669   \edef\@do@newglossaryentry{%
6670     \noexpand\newglossaryentry{\the\glslabeltok}%
6671     {%
6672       type=\acronymtype,%
6673       name={\noexpand\acronymfont{\the\glsshorttok}},%
6674       sort={\the\glsshorttok},%
6675       text={\the\glsshorttok},%

```

Default to the short plural.

```

6676     plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6677     first={\the\glslongtok},%

```

Default to the long plural.

```

6678     firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6679     short={\the\glsshorttok},%
6680     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6681     long={\the\glslongtok},%
6682     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6683     description={\noexpand\@glo@first},%
6684     descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6685     symbol={\the\glsshorttok},%

```

Default to the short plural.

```

6686     symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6687     \the\glskeylisttok
6688   }%
6689 }%
6690 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6691 \let\@org@gls@assign@plural\gls@assign@plural
6692 \let\@org@gls@assign@descplural\gls@assign@descplural
6693 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6694 \def\gls@assign@firstpl##1##2{%
6695   \@@gls@expand@field{##1}{firstpl}{##2}%
6696 }%
6697 \def\gls@assign@plural##1##2{%
6698   \@@gls@expand@field{##1}{plural}{##2}%
6699 }%
6700 \def\gls@assign@descplural##1##2{%
6701   \@@gls@expand@field{##1}{descplural}{##2}%
6702 }%
6703 \def\gls@assign@symbolplural##1##2{%
6704   \@@gls@expand@field{##1}{symbolplural}{##2}%
6705 }%
6706 \@do@newglossaryentry
6707 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6708 \let\gls@assign@plural\@org@gls@assign@plural
6709 \let\gls@assign@descplural\@org@gls@assign@descplural
6710 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6711 }

```

`\setSmallAcronymStyle` Neither footnote nor description required, but smallcaps or smaller specified.  
Use the symbol key to store the short form and first to store the long form.

```

6712 \newcommand*{\SetSmallAcronymStyle}{%
6713   \renewcommand{\newacronym}[4][[]]{%
6714     \ifx\@glsacronymlists\@empty
6715       \def\@glo@type{\acronymtype}%
6716       \setkeys{glossentry}{##1}%
6717       \DeclareAcronymList{\@glo@type}%
6718       \SetSmallAcronymDisplayStyle{\@glo@type}%
6719     \fi
6720     \glskeylisttok{##1}%
6721     \glslabeltok{##2}%
6722     \glsshorttok{##3}%
6723     \glslongtok{##4}%
6724     \newacronymhook
6725     \SmallNewAcronymDef
6726   }%

```

Change the display since first only contains long form.

```

6727   \@for\@gls@type:=\@glsacronymlists\do{%
6728     \SetSmallAcronymDisplayStyle{\@gls@type}%

```

```
6729 }%
```

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.

```
6730 \ifglsmallcaps
6731   \renewcommand*{\acronymfont}[1]{\textsc{##1}}
6732   \renewcommand*{\acrpluralsuffix}{%
6733     \glstextup{\glspluralsuffix}}%
6734 \else
6735   \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}
6736 \fi
```

check for option clash

```
6737 \ifglsmallcaps
6738   \ifglsmallcaps
6739     \PackageError{glossaries}{Option clash: 'smallcaps' and 'dua'
6740       can't both be set}{}%
6741   \else
6742     \PackageError{glossaries}{Option clash: 'smaller' and 'dua'
6743       can't both be set}{}%
6744 \fi
6745 \fi
6746 }%
```

\SetDUADisplayStyle Sets the acronym display style for given glossary with dua setting.

```
6747 \newcommand*{\SetDUADisplayStyle}[1]{%
6748   \defglentryfmt[##1]{\glsgenentryfmt}%
6749 }
```

\DUANewAcronymDef

```
6750 \newcommand*{\DUANewAcronymDef}{%
6751   \edef\@do@newglossaryentry{%
6752     \noexpand\newglossaryentry{\the\glslabeltok}%
6753     {%
6754       type=\acronymtype,%
6755       name={\the\glsshorttok},%
6756       text={\the\glslongtok},%
6757       first={\the\glslongtok},%
6758       plural={\noexpand\expandonce\noexpand\@glo@longpl},%
6759       firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6760       short={\the\glsshorttok},%
6761       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6762       long={\the\glslongtok},%
6763       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6764       description={\the\glslongtok},%
6765       descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6766       symbol={\the\glsshorttok},%
6767       symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6768       \the\glskeylisttok
```



```

6769 }%
6770 }%
6771 \let\@org@gl@s@assign@firstpl\gl@s@assign@firstpl
6772 \let\@org@gl@s@assign@plural\gl@s@assign@plural
6773 \let\@org@gl@s@assign@symbolplural\gl@s@assign@symbolplural
6774 \let\@org@gl@s@assign@descplural\gl@s@assign@descplural
6775 \def\gl@s@assign@firstpl##1##2{%
6776   \@@gl@s@expand@field{##1}{firstpl}{##2}%
6777 }%
6778 \def\gl@s@assign@plural##1##2{%
6779   \@@gl@s@expand@field{##1}{plural}{##2}%
6780 }%
6781 \def\gl@s@assign@symbolplural##1##2{%
6782   \@@gl@s@expand@field{##1}{symbolplural}{##2}%
6783 }%
6784 \def\gl@s@assign@descplural##1##2{%
6785   \@@gl@s@expand@field{##1}{descplural}{##2}%
6786 }%
6787 \do@newglossaryentry
6788 \let\gl@s@assign@firstpl\@org@gl@s@assign@firstpl
6789 \let\gl@s@assign@plural\@org@gl@s@assign@plural
6790 \let\gl@s@assign@symbolplural\@org@gl@s@assign@symbolplural
6791 \let\gl@s@assign@descplural\@org@gl@s@assign@descplural
6792 }

```

\SetDUASyle Always expand acronyms.

```

6793 \newcommand*\SetDUASyle{%
6794   \renewcommand{\newacronym}[4][\]{%
6795     \ifx\@gl@s@acronymlists\@empty
6796       \def\@glo@type{\acronymtype}%
6797       \setkeys{glossentry}{##1}%
6798       \DeclareAcronymList{\@glo@type}%
6799       \SetDUADisplayStyle{\@glo@type}%
6800     \fi
6801     \gl@s@keylisttok{##1}%
6802     \gl@s@labeltok{##2}%
6803     \gl@s@shorttok{##3}%
6804     \gl@s@longtok{##4}%
6805     \newacronymhook
6806     \DUANewAcronymDef
6807   }%
6808   \SetDUADisplayStyle{\@glo@type}%
6809   \SetDUADisplayStyle{\@glo@type}%
6810 }%
6811 }

```

\SetAcronymStyle

```

6812 \newcommand*\SetAcronymStyle{%
6813   \SetDefaultAcronymStyle
6814   \ifglssacrdescription
6815     \ifglssacrfootnote
6816       \SetDescriptionFootnoteAcronymStyle
6817     \else
6818       \ifglssacrdua
6819         \SetDescriptionDUAAcronymStyle
6820       \else
6821         \SetDescriptionAcronymStyle
6822       \fi
6823     \fi
6824   \else
6825     \ifglssacrfootnote
6826       \SetFootnoteAcronymStyle
6827     \else
6828       \ifthenelse{\boolean{glssacrsmalldcaps}}{\OR
6829         \boolean{glssacrsmaller}}}%
6830     {%
6831       \SetSmallAcronymStyle
6832     }%
6833     {%
6834       \ifglssacrdua
6835         \SetDUASStyle
6836       \fi
6837     }%
6838   \fi
6839 \fi
6840 }

```

Set the acronym style according to the package options

```
6841 \SetAcronymStyle
```

Allow user to define their own custom acronyms. (For compatibility with versions before v3.0, the short form is stored in the user1 key, the plural short form is stored in the user2 key, the long form is stored in the user3 key and the plural long form is stored in the user4 key.) Defaults to displaying only the acronym with the long form as the description.

**CustomDisplayStyle** Sets the acronym display style.

```

6842 \newcommand*\SetCustomDisplayStyle[1]{%
6843   \defglssentryfmt[#1]{\glsgenentryfmt}%
6844 }

```

**CustomAcronymFields**

```

6845 \newcommand*\CustomAcronymFields{%
6846   name={\the\glsshorttok},%
6847   description={\the\glslongtok},%
6848   first={\acrfullformat{\the\glslongtok}{\the\glsshorttok}},%

```

```

6849 firstplural={\acrfullformat
6850   {\noexpand\glsentrylongpl{\the\glslabeltok}}}%
6851   {\noexpand\glsentryshortpl{\the\glslabeltok}}},%

6852 text={\the\glsshorttok},%
6853 plural={\the\glsshorttok\noexpand\acrpluralsuffix}%
6854 }

```

#### CustomNewAcronymDef

```

6855 \newcommand*{\CustomNewAcronymDef}{%
6856   \protected@edef\@do@newglossaryentry{%
6857     \noexpand\newglossaryentry{\the\glslabeltok}%
6858     {%
6859       type=\acronymtype,%
6860       short={\the\glsshorttok},%
6861       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6862       long={\the\glslongtok},%
6863       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6864       user1={\the\glsshorttok},%
6865       user2={\the\glsshorttok\noexpand\acrpluralsuffix},%
6866       user3={\the\glslongtok},%
6867       user4={\the\glslongtok\noexpand\acrpluralsuffix},%
6868       \CustomAcronymFields,%
6869       \the\glskeylisttok
6870     }%
6871   }%
6872   \@do@newglossaryentry
6873 }

```

#### \SetCustomStyle

```

6874 \newcommand*{\SetCustomStyle}{%
6875   \renewcommand{\newacronym}[4][\]{%
6876     \ifx\@glsacronymlists\@empty
6877       \def\@glo@type{\acronymtype}%
6878       \setkeys{glossentry}{##1}%
6879       \DeclareAcronymList{\@glo@type}%
6880       \SetCustomDisplayStyle{\@glo@type}%
6881     \fi
6882     \glskeylisttok{##1}%
6883     \glslabeltok{##2}%
6884     \glsshorttok{##3}%
6885     \glslongtok{##4}%
6886     \newacronymhook
6887     \CustomNewAcronymDef
6888   }%

   Set the display
6889   \@for\@gls@type:=\@glsacronymlists\do{%
6890     \SetCustomDisplayStyle{\@gls@type}%
6891   }%

```

6892 }

## 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.

```
6893 \RequirePackage{glossary-hypernav}
```

The styles that use list-like environments. These are not loaded if the `nolist` option is used:

```
6894 \@gls@loadlist
```

The styles that use the `longtable` environment. These are not loaded if the `no-long package` option is used.

```
6895 \@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.

```
6896 \@gls@loadsuper
```

The tree-like styles. These are not loaded if the `notree` package option is used.

```
6897 \@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.

```
6898 \ifx\@glossary@default@style\relax
6899 \else
6900   \setglossarystyle{\@glossary@default@style}
6901 \fi
```

## 1.19 Debugging Commands

`\showgloparent`    `\showgloparent{<label>}`

```
6902 \newcommand*{\showgloparent}[1]{%
6903   \expandafter\show\csname glo@\glsdetoklabel{#1}@parent\endcsname
6904 }
```

`\showglolevel`    `\showglolevel{<label>}`

```
6905 \newcommand*{\showglolevel}[1]{%
6906   \expandafter\show\csname glo@\glsdetoklabel{#1}@level\endcsname
6907 }
```

`\showglotext` `\showglotext{<label>}`

```
6908 \newcommand*{\showglotext}[1]{%
6909   \expandafter\show\csname glo@glstdetoklabel{#1}@text\endcsname
6910 }
```

`\showgloplural` `\showgloplural{<label>}`

```
6911 \newcommand*{\showgloplural}[1]{%
6912   \expandafter\show\csname glo@glstdetoklabel{#1}@plural\endcsname
6913 }
```

`\showglofirst` `\showglofirst{<label>}`

```
6914 \newcommand*{\showglofirst}[1]{%
6915   \expandafter\show\csname glo@glstdetoklabel{#1}@first\endcsname
6916 }
```

`\showglofirstpl` `\showglofirstpl{<label>}`

```
6917 \newcommand*{\showglofirstpl}[1]{%
6918   \expandafter\show\csname glo@glstdetoklabel{#1}@firstpl\endcsname
6919 }
```

`\showglotype` `\showglotype{<label>}`

```
6920 \newcommand*{\showglotype}[1]{%
6921   \expandafter\show\csname glo@glstdetoklabel{#1}@type\endcsname
6922 }
```

`\showglocounter` `\showglocounter{<label>}`

```
6923 \newcommand*{\showglocounter}[1]{%
6924   \expandafter\show\csname glo@glstdetoklabel{#1}@counter\endcsname
6925 }
```

\showglouserii    \showglouserii{\label{}}

```
6926 \newcommand*{\showglouserii}[1]{%
6927   \expandafter\show\csname glo@\glstetoklabel{#1}@userii\endcsname
6928 }
```

\showglouseriii    \showglouseriii{\label{}}

```
6929 \newcommand*{\showglouseriii}[1]{%
6930   \expandafter\show\csname glo@\glstetoklabel{#1}@useriii\endcsname
6931 }
```

\showglouseriv    \showglouseriv{\label{}}

```
6932 \newcommand*{\showglouseriv}[1]{%
6933   \expandafter\show\csname glo@\glstetoklabel{#1}@useriv\endcsname
6934 }
```

\showglouserv    \showglouserv{\label{}}

```
6935 \newcommand*{\showglouserv}[1]{%
6936   \expandafter\show\csname glo@\glstetoklabel{#1}@userv\endcsname
6937 }
```

\showglouservi    \showglouservi{\label{}}

```
6938 \newcommand*{\showglouservi}[1]{%
6939   \expandafter\show\csname glo@\glstetoklabel{#1}@uservi\endcsname
6940 }
```

\showglouservi    \showglouservi{\label{}}

```
6941 \newcommand*{\showglouservi}[1]{%
6942   \expandafter\show\csname glo@\glstetoklabel{#1}@uservi\endcsname
6943 }
```

`\showglo`name `\showglo`name{<label>}

```
6944 \newcommand*{\showglo
```

name}[1]{%  
6945 \expandafter\show\csname glo@\glsdetoklabel{#1}@name\endcsname  
6946 }

`\showglo`desc `\showglo`desc{<label>}

```
6947 \newcommand*{\showglo
```

desc}[1]{%  
6948 \expandafter\show\csname glo@\glsdetoklabel{#1}@desc\endcsname  
6949 }

`\showglo`descplural `\showglo`descplural{<label>}

```
6950 \newcommand*{\showglo
```

descplural}[1]{%  
6951 \expandafter\show\csname glo@\glsdetoklabel{#1}@descplural\endcsname  
6952 }

`\showglo`sort `\showglo`sort{<label>}

```
6953 \newcommand*{\showglo
```

sort}[1]{%  
6954 \expandafter\show\csname glo@\glsdetoklabel{#1}@sort\endcsname  
6955 }

`\showglo`symbol `\showglo`symbol{<label>}

```
6956 \newcommand*{\showglo
```

symbol}[1]{%  
6957 \expandafter\show\csname glo@\glsdetoklabel{#1}@symbol\endcsname  
6958 }

`\showglo`symbolplural `\showglo`symbolplural{<label>}

```
6959 \newcommand*{\showglo
```

symbolplural}[1]{%  
6960 \expandafter\show\csname glo@\glsdetoklabel{#1}@symbolplural\endcsname  
6961 }

`\showgloshort` `\showgloshort{<label>}`

```
6962 \newcommand*{\showgloshort}[1]{%
6963   \expandafter\show\csname glo@\glsdetoklabel{#1}@short\endcsname
6964 }
```

`\showglolong` `\showglolong{<label>}`

```
6965 \newcommand*{\showglolong}[1]{%
6966   \expandafter\show\csname glo@\glsdetoklabel{#1}@long\endcsname
6967 }
```

`\showgloindex` `\showgloindex{<label>}`

```
6968 \newcommand*{\showgloindex}[1]{%
6969   \expandafter\show\csname glo@\glsdetoklabel{#1}@index\endcsname
6970 }
```

`\showgloflag` `\showgloflag{<label>}`

```
6971 \newcommand*{\showgloflag}[1]{%
6972   \expandafter\show\csname ifglo@\glsdetoklabel{#1}@flag\endcsname
6973 }
```

`\showgloloclist` `\showgloloclist{<label>}`

```
6974 \newcommand*{\showgloloclist}[1]{%
6975   \expandafter\show\csname glo@\glsdetoklabel{#1}@loclist\endcsname
6976 }
```

`\showacronymlists` `\showacronymlists`

Show list of glossaries that have been flagged as a list of acronyms.

```
6977 \newcommand*{\showacronymlists}{%
6978   \show\@glsacronymlists
6979 }
```



`\showglossaries` `\showglossaries`

Show list of defined glossaries.

```
6980 \newcommand*{\showglossaries}{%
6981   \show\@glo@types
6982 }
```

`\showglossaryin` `\showglossaryin{<glossary-label>}`

Show the ‘in’ extension for the given glossary.

```
6983 \newcommand*{\showglossaryin}[1]{%
6984   \expandafter\show\csname @glotype@#1@in\endcsname
6985 }
```

`\showglossaryout` `\showglossaryout{<glossary-label>}`

Show the ‘out’ extension for the given glossary.

```
6986 \newcommand*{\showglossaryout}[1]{%
6987   \expandafter\show\csname @glotype@#1@out\endcsname
6988 }
```

`\showglossarytitle` `\showglossarytitle{<glossary-label>}`

Show the title for the given glossary.

```
6989 \newcommand*{\showglossarytitle}[1]{%
6990   \expandafter\show\csname @glotype@#1@title\endcsname
6991 }
```

`\showglossarycounter` `\showglossarycounter{<glossary-label>}`

Show the counter for the given glossary.

```
6992 \newcommand*{\showglossarycounter}[1]{%
6993   \expandafter\show\csname @glotype@#1@counter\endcsname
6994 }
```

`\showglossaryentries` `\showglossaryentries{<glossary-label>}`

Show the list of entry labels for the given glossary.

```
6995 \newcommand*{\showglossaryentries}[1]{%
6996   \expandafter\show\csname glolist@#1\endcsname
6997 }
```

## 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.

```
6998 \csname ifglscompatible-2.07\endcsname
6999 \RequirePackage{glossaries-compatible-207}
7000 \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.

```
7001 \NeedsTeXFormat{LaTeX2e}
7002 \ProvidesPackage{glossaries-prefix}[2014/07/30 v4.08 (NLCT)]

Pass all options to glossaries:
7003 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{glossaries}}

Process options:
7004 \ProcessOptions

Load glossaries:
7005 \RequirePackage{glossaries}

Add the new keys:
7006 \define@key{glossentry}{prefixfirst}{\def\@glo@entryprefixfirst{#1}}%
7007 \define@key{glossentry}{prefixfirstplural}{\def\@glo@entryprefixfirstplural{#1}}%
7008 \define@key{glossentry}{prefix}{\def\@glo@entryprefix{#1}}%
7009 \define@key{glossentry}{prefixplural}{\def\@glo@entryprefixplural{#1}}%

Add them to \@gls@keymap:
7010 \appto\@gls@keymap{,%
7011   {prefixfirst}{prefixfirst},%
7012   {prefixfirstplural}{prefixfirstplural},%
7013   {prefix}{prefix},%
7014   {prefixplural}{prefixplural}}%
7015 }
```

Set the default values:

```

7016 \appto\@newglossaryentryprehook{%
7017   \def\@glo@entryprefix{}%
7018   \def\@glo@entryprefixplural{}%
7019   \let\@glo@entryprefixfirst\@gls@default@value
7020   \let\@glo@entryprefixfirstplural\@gls@default@value
7021 }

```

Set the assignment code:

```

7022 \appto\@newglossaryentryposthook{%
7023   \gls@assign@field{\@glo@label}{prefix}{\@glo@entryprefix}%
7024   \gls@assign@field{\@glo@label}{prefixplural}{\@glo@entryprefixplural}%

```

If prefixfirst has not been supplied, make it the same as prefix.

```

7025   \expandafter\gls@assign@field\expandafter
7026     {\csname glo@\@glo@label @prefix\endcsname}{\@glo@label}{prefixfirst}%
7027     {\@glo@entryprefixfirst}%

```

If prefixfirstplural has not been supplied, make it the same as prefixplural.

```

7028   \expandafter\gls@assign@field\expandafter
7029     {\csname glo@\@glo@label @prefixplural\endcsname}{\@glo@label}%
7030     {prefixfirstplural}{\@glo@entryprefixfirstplural}%
7031 }

```

Define commands to access these fields:

glsentryprefixfirst

```

7032 \newcommand*\glsentryprefixfirst[1]{\csuse{glo@#1@prefixfirst}}

```

ryprefixfirstplural

```

7033 \newcommand*\glsentryprefixfirstplural[1]{\csuse{glo@#1@prefixfirstplural}}

```

\glsentryprefix

```

7034 \newcommand*\glsentryprefix[1]{\csuse{glo@#1@prefix}}

```

lsentryprefixplural

```

7035 \newcommand*\glsentryprefixplural[1]{\csuse{glo@#1@prefixplural}}

```

Now for the initial upper case variants:

Glsentryprefixfirst

```

7036 \newrobustcmd*\Glsentryprefixfirst[1]{%
7037   \protected@edef\@glo@text{\csname glo@#1@prefixfirst\endcsname}%
7038   \xmakefirstuc\@glo@text
7039 }

```

ryprefixfirstplural

```

7040 \newrobustcmd*\Glsentryprefixfirstplural[1]{%
7041   \protected@edef\@glo@text{\csname glo@#1@prefixfirstplural\endcsname}%
7042   \xmakefirstuc\@glo@text
7043 }

```

\Glsentryprefix

```
7044 \newrobustcmd*{\Glsentryprefix}[1]{%
7045   \protected@edef\@glo@text{\csname glo@#1@prefix\endcsname}%
7046   \xmakefirstuc\@glo@text
7047 }
```

lentryprefixplural

```
7048 \newrobustcmd*{\Glsentryprefixplural}[1]{%
7049   \protected@edef\@glo@text{\csname glo@#1@prefixplural\endcsname}%
7050   \xmakefirstuc\@glo@text
7051 }
```

Define commands to determine if the prefix keys have been set:

\ifglshasprefix

```
7052 \newcommand*{\ifglshasprefix}[3]{%
7053   \ifcempty{glo@#1@prefix}%
7054   {#3}%
7055   {#2}%
7056 }
```

ifglshasprefixplural

```
7057 \newcommand*{\ifglshasprefixplural}[3]{%
7058   \ifcempty{glo@#1@prefixplural}%
7059   {#3}%
7060   {#2}%
7061 }
```

ifglshasprefixfirst

```
7062 \newcommand*{\ifglshasprefixfirst}[3]{%
7063   \ifcempty{glo@#1@prefixfirst}%
7064   {#3}%
7065   {#2}%
7066 }
```

asprefixfirstplural

```
7067 \newcommand*{\ifglshasprefixfirstplural}[3]{%
7068   \ifcempty{glo@#1@prefixfirstplural}%
7069   {#3}%
7070   {#2}%
7071 }
```

Define commands that insert the prefix before commands like \gls:

\pgls

```
7072 \newrobustcmd{\pgls}{\@gls@hyp@opt\@pgls}
```

\@pgls Unstarred version.

```
7073 \newcommand*\@pgls}[2] [] {%
7074   \new@ifnextchar [%
7075     {\@pgls@{#1}{#2}}%
7076     {\@pgls@{#1}{#2} []}%
7077 }
```

\@pgls@ Read in the final optional argument:

```
7078 \def\@pgls@#1#2[#3] {%
7079   \glsdoifexists{#2}%
7080   {%
7081     \ifglsused{#2}%
7082     {%
7083       \glsentryprefix{#2}%
7084     }%
7085     {%
7086       \glsentryprefixfirst{#2}%
7087     }%
7088     \@gls@{#1}{#2}[#3]%
7089   }%
7090 }
```

Similarly for the plural version:

\pglsp1

```
7091 \newrobustcmd{\pglsp1}{\@gls@hyp@opt\@pglsp1}
```

\@pglsp1 Unstarred version.

```
7092 \newcommand*\@pglsp1}[2] [] {%
7093   \new@ifnextchar [%
7094     {\@pglsp1@{#1}{#2}}%
7095     {\@pglsp1@{#1}{#2} []}%
7096 }
```

\@pglsp1@ Read in the final optional argument:

```
7097 \def\@pglsp1@#1#2[#3] {%
7098   \glsdoifexists{#2}%
7099   {%
7100     \ifglsused{#2}%
7101     {%
7102       \glsentryprefixplural{#2}%
7103     }%
7104     {%
7105       \glsentryprefixfirstplural{#2}%
7106     }%
7107     \@glspl@{#1}{#2}[#3]%
7108   }%
7109 }
```

Now for the first letter upper case versions:

`\PglS`

```
7110 \newrobustcmd{\PglS}{\@gls@hyp@opt\PglS}
```

`\@PglS` Unstarred version.

```
7111 \newcommand*{\@PglS}[2][\%  
7112 \new@ifnextchar[%  
7113 {\@PglS@{#1}{#2}}%  
7114 {\@PglS@{#1}{#2}[]}%  
7115 }
```

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

```
7116 \def\@PglS@#1#2[#3]{%  
7117 \glsdoifexists{#2}%  
7118 {%  
7119 \ifglSused{#2}%  
7120 {%  
7121 \ifglshasprefix{#2}%  
7122 {%  
7123 \Glsentryprefix{#2}%  
7124 \@gls@{#1}{#2}[#3]%  
7125 }%  
7126 {\@Gls@{#1}{#2}[#3]}%  
7127 }%  
7128 {%  
7129 \ifglshasprefixfirst{#2}%  
7130 {%  
7131 \Glsentryprefixfirst{#2}%  
7132 \@gls@{#1}{#2}[#3]%  
7133 }%  
7134 {\@Gls@{#1}{#2}[#3]}%  
7135 }%  
7136 }%  
7137 }
```

Similarly for the plural version:

`\PglSpl`

```
7138 \newrobustcmd{\PglSpl}{\@gls@hyp@opt\PglSpl}
```

`\@PglSpl` Unstarred version.

```
7139 \newcommand*{\@PglSpl}[2][\%  
7140 \new@ifnextchar[%  
7141 {\@PglSpl@{#1}{#2}}%  
7142 {\@PglSpl@{#1}{#2}[]}%  
7143 }
```

\@Pglsp1@ Read in the final optional argument:

```
7144 \def\@Pglsp1@#1#2[#3]{%
7145   \glsdoifexists{#2}%
7146   {%
7147     \ifglsused{#2}%
7148     {%
7149       \ifglshasprefixplural{#2}%
7150       {%
7151         \Glsentryprefixplural{#2}%
7152         \@glsp1@{#1}{#2}[#3]%
7153       }%
7154       {\@Glspl@{#1}{#2}[#3]}%
7155     }%
7156     {%
7157       \ifglshasprefixfirstplural{#2}%
7158       {%
7159         \Glsentryprefixfirstplural{#2}%
7160         \@glsp1@{#1}{#2}[#3]%
7161       }%
7162       {\@Glspl@{#1}{#2}[#3]}%
7163     }%
7164   }%
7165 }
```

Finally the all upper case versions:

\PGLS

```
7166 \newrobustcmd{\PGLS}{\@gls@hyp@opt\@PGLS}
```

\@PGLS Unstarred version.

```
7167 \newcommand*{\@PGLS}[2][ ]{%
7168   \new@ifnextchar[%
7169   {\@PGLS@{#1}{#2}}%
7170   {\@PGLS@{#1}{#2}[ ]}%
7171 }
```

\@PGLS@ Read in the final optional argument:

```
7172 \def\@PGLS@#1#2[#3]{%
7173   \glsdoifexists{#2}%
7174   {%
7175     \ifglsused{#2}%
7176     {%
7177       \mfirstucMakeUppercase{\glsentryprefix{#2}}%
7178     }%
7179     {%
7180       \mfirstucMakeUppercase{\glsentryprefixfirst{#2}}%
7181     }%
7182     \@GLS@{#1}{#2}[#3]%

```

```

7183 }%
7184 }

```

Plural version:

`\PGLSp1`

```

7185 \newrobustcmd{\PGLSp1}{\@gls@hyp@opt\PGLSp1}

```

`\@PGLSp1` Unstarred version.

```

7186 \newcommand*{\@PGLSp1}[2][{}]{%
7187   \new@ifnextchar[%
7188     {\@PGLSp1@{#1}{#2}}%
7189     {\@PGLSp1@{#1}{#2}[]}%
7190 }

```

`\@PGLSp1@` Read in the final optional argument:

```

7191 \def\@PGLSp1@#1#2[#3]{%
7192   \glsdoifexists{#2}%
7193   {%
7194     \ifglsused{#2}%
7195     {%
7196       \mfirstucMakeUppercase{\glsentryprefixplural{#2}}%
7197     }%
7198     {%
7199       \mfirstucMakeUppercase{\glsentryprefixfirstplural{#2}}%
7200     }%
7201     \@GLSp1@{#1}{#2}[#3]%
7202   }%
7203 }

```

### 3 Mfirstuc Documented Code

```

7204 \NeedsTeXFormat{LaTeX2e}
7205 \ProvidesPackage{mfirstuc}[2014/07/30 v1.09 (NLCT)]

```

Requires etoolbox:

```

7206 \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`.

```

7207 \newif\if@glscs

```



```

7208 \newtoks\@glsmfirst
7209 \newtoks\@glsmrest
7210 \newrobustcmd*{\makefirstuc}[1]{%
7211   \def\@gls@argi{#1}%
7212   \ifx\@gls@argi\@empty
       If the argument is empty, do nothing.
7213   \else

7214     \def\@gls@tmp{\ #1}%
7215     \@onelevel@sanitize\@gls@tmp
7216     \expandafter\@gls@checkcs\@gls@tmp\relax\relax
7217     \if@glscs
7218       \@gls@getbody #1{}\@nil
7219       \ifx\@gls@rest\@empty
7220         \glsmakefirstuc{#1}%
7221       \else
7222         \expandafter\@gls@split\@gls@rest\@nil
7223         \ifx\@gls@first\@empty
7224           \glsmakefirstuc{#1}%
7225         \else
7226           \expandafter\@glsmfirst\expandafter{\@gls@first}%
7227           \expandafter\@glsmrest\expandafter{\@gls@rest}%
7228           \edef\@gls@domfirstuc{\noexpand\@gls@body
7229             {\noexpand\glsmakefirstuc\the\@glsmfirst}%
7230             \the\@glsmrest}%
7231           \@gls@domfirstuc
7232         \fi
7233       \fi
7234     \else
7235       \glsmakefirstuc{#1}%
7236     \fi
7237   \fi
7238 }

       Put first argument in \@gls@first and second argument in \@gls@rest:
7239 \def\@gls@split#1#2\@nil{%
7240   \def\@gls@first{#1}\def\@gls@rest{#2}%
7241 }

7242 \def\@gls@checkcs#1 #2#3\relax{%
7243   \def\@gls@argi{#1}\def\@gls@argii{#2}%
7244   \ifx\@gls@argi\@gls@argii
7245     \@glscstrue
7246   \else
7247     \@glscsfalse
7248   \fi
7249 }

```

\@gls@makefirstuc    Make first thing upper case:

```

7250 \def\@gls@makefirstuc#1{\mfirstucMakeUppercase #1}

```

irstucMakeUppercase Allow user to replace \MakeUppercase with another case changing command.

```
7251 \newcommand*{\mfirstucMakeUppercase}{\MakeUppercase}
```

\glsmakefirstuc Provide a user command to make it easier to customise.

```
7252 \newcommand*{\glsmakefirstuc}[1]{\@gls@makefirstuc{#1}}
```

Get the first grouped argument and stores in \@gls@body.

```
7253 \def\@gls@getbody#1#\def\@gls@body{#1}\@gls@gobbletonil}
```

Scoop up everything to \@nil and store in \@gls@rest:

```
7254 \def\@gls@gobbletonil#1\@nil{\def\@gls@rest{#1}}
```

\xmakefirstuc Expand argument once before applying \makefirstuc (added v1.01).

```
7255 \newcommand*{\xmakefirstuc}[1]{%
```

```
7256 \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).

```
7257 \newrobustcmd*{\capitalisewords}[1]{%
```

```
7258 \def\gls@add@space{}%
```

```
7259 \let\@mfu@domakefirstuc\makefirstuc
```

```
7260 \let\@mfu@checkword\@gobble
```

```
7261 \mfu@capitalisewords#1 \@nil\mfu@endcap
```

```
7262 }
```

```
7263 \def\mfu@capitalisewords#1 #2\mfu@endcap{%
```

```
7264 \def\mfu@cap@first{#1}%
```

```
7265 \def\mfu@cap@second{#2}%
```

```
7266 \gls@add@space
```

```
7267 \@mfu@checkword{#1}%
```

```
7268 \@mfu@domakefirstuc{#1}%
```

```
7269 \def\gls@add@space{ }%
```

```
7270 \ifx\mfu@cap@second\@nnil
```

```
7271 \let\next@mfu@cap\mfu@noop
```

```
7272 \else
```

```
7273 \let\next@mfu@cap\mfu@capitalisewords
```

```
7274 \let\@mfu@checkword\mfu@checkword
```

```
7275 \fi
```

```
7276 \next@mfu@cap#2\mfu@endcap
```

```
7277 }
```

```
7278 \def\mfu@noop#1\mfu@endcap{}
```

\mfu@checkword Check if word should be capitalised.

```
7279 \newcommand*{\mfu@checkword}[1]{%
```

```
7280 \ifinlist{#1}{\@mfu@nocaplist}%
```

```
7281 {%
```

```
7282 \let\@mfu@domakefirstuc\@firstofone
```

```
7283 }%
```

```
7284 {%
```

```

7285 \let\@mfu@domakefirstuc\makefirstuc
7286 }%
7287 }

```

\@mfu@nocaplist List of words that shouldn't be capitalised.

```
7288 \newcommand*\@mfu@nocaplist{}
```

\MFUnocap Provide the user with a means to add a word to the list.

```
7289 \newcommand*\MFUnocap[1]{\listadd{\@mfu@nocaplist}{#1}}
```

\gMFUnocap Global version.

```
7290 \newcommand*\gMFUnocap[1]{\listgadd{\@mfu@nocaplist}{#1}}
```

\MFUclear Clear the list

```
7291 \newcommand*\MFUclear{\renewcommand*\@mfu@nocaplist{}}
```

\xcapitalisewords Short-cut command:

```

7292 \newcommand*\xcapitalisewords[1]{%
7293 \expandafter\capitalisewords\expandafter{#1}%
7294 }

```

## 4 Mfirstuc-english Documented Code

```

7295 \NeedsTeXFormat{LaTeX2e}
7296 \ProvidesPackage{mfirstuc-english}[2014/07/30 v1.0 (NLCT)]

```

Load mfirstuc if not already loaded:

```
7297 \RequirePackage{mfirstuc}
```

Add no-cap words. (List isn't a complete list.)

```

7298 \MFUnocap{a}
7299 \MFUnocap{an}
7300 \MFUnocap{and}
7301 \MFUnocap{but}
7302 \MFUnocap{for}
7303 \MFUnocap{in}
7304 \MFUnocap{of}
7305 \MFUnocap{or}
7306 \MFUnocap{no}
7307 \MFUnocap{nor}
7308 \MFUnocap{so}
7309 \MFUnocap{some}
7310 \MFUnocap{the}
7311 \MFUnocap{with}
7312 \MFUnocap{yet}

```

## 5 Glossary Styles

### 5.1 Glossary hyper-navigation definitions (glossary-hypernav package)

Package Definition:

```
7313 \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`

```
7314 \newcommand*{\glsnavhyperlink}[3][\@glo@type]{%
7315   \edef\gls@grplabel{#2}\protected@edef\gls@grptitle{#3}%
7316   \@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`

```
7317 \newcommand*{\glsnavhypertarget}[3][\@glo@type]{%
  Add this group to the aux file for re-run check.
7318   \protected@write\@auxout{}{\string\@gls@hypergroup{#1}{#2}}%
  Add the target.
7319   \@glstarget{glsn:#1@#2}{#3}%
  Check list of know groups to determine if a re-run is required.
7320   \expandafter\let
7321     \expandafter\@gls@list\csname @gls@hypergroup@#1\endcsname
  Iterate through list and terminate loop if this group is found.
7322   \@for\@gls@elem:=\@gls@list\do{%
7323     \ifthenelse{\equal{\@gls@elem}{#2}}{\@endfortrue}{}%
  Check if list terminated prematurely.
7324   \if@endfor
7325   \else
```

This group was not included in the list, so issue a warning.

```

7326 \GlossariesWarningNoLine{Navigation panel
7327   for glossary type ‘#1’^^Jmissing group ‘#2’}%
7328 \gdef\gls@hypergroupprerun{%
7329   \GlossariesWarningNoLine{Navigation panel
7330   has changed. Rerun LaTeX}}%
7331 \fi
7332 }

```

`\gls@hypergroupprerun` Give a warning at the end if re-run required

```

7333 \let\gls@hypergroupprerun\relax
7334 \AtEndDocument{\gls@hypergroupprerun}

```

`\@gls@hypergroup` This adds to (or creates) the command `\@gls@hypergrouplist@<glossary type>` which lists all groups for a given glossary, so that the navigation bar only contains those groups that are present. However it requires at least 2 runs to ensure the information is up-to-date.

```

7335 \newcommand*{\@gls@hypergroup}[2]{%
7336 \@ifundefined{\@gls@hypergrouplist@#1}{%
7337   \expandafter\xdef\csname @gls@hypergrouplist@#1\endcsname{#2}%
7338 }{%
7339   \expandafter\let\expandafter\@gls@tmp
7340   \csname @gls@hypergrouplist@#1\endcsname
7341   \expandafter\xdef\csname @gls@hypergrouplist@#1\endcsname{%
7342     \@gls@tmp,#2}%
7343 }%
7344 }

```

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`

```

7345 \newcommand*{\glsnavigation}{%
7346 \def\@gls@between{}%
7347 \@ifundefined{\@gls@hypergrouplist@\@glo@type}{%
7348   \def\@gls@list{}%
7349 }{%
7350   \expandafter\let\expandafter\@gls@list
7351   \csname @gls@hypergrouplist@\@glo@type\endcsname
7352 }%
7353 \@for\@gls@tmp:=\@gls@list\do{%
7354   \@gls@between
7355   \@gls@getgrouptitle{\@gls@tmp}{\@gls@grptitle}%

```

```

7356 \glsnavhyperlink{\@gls@tmp}{\@gls@grptitle}%
7357 \let\@gls@between\glshypernavsep%
7358 }%
7359 }

```

`\glshypernavsep` Separator for the hyper navigation bar.

```

7360 \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`

```

7361 \newcommand*{\glssymbolnav}{%
7362 \glsnavhyperlink{glssymbols}{\glsgetgrouptitle{glssymbols}}%
7363 \glshypernavsep
7364 \glsnavhyperlink{glsnumbers}{\glsgetgrouptitle{glsnumbers}}%
7365 \glshypernavsep
7366 }

```

## 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.

```

7367 \ProvidesPackage{glossary-inline}[2013/11/14 v4.0 (NLCT)]

```

`inline` Define the inline style.

```

7368 \newglossarystyle{inline}{%

```

Start of glossary sets up first empty separator between entries. (This is then changed by `\glossentry`)

```

7369 \renewenvironment{theglossary}%
7370 {%
7371 \def\gls@inlinesep{}%
7372 \def\gls@inlinesubsep{}%
7373 \def\gls@inlinepostchild{}%
7374 }%
7375 {\glspostinline}%

```

No header:

```

7376 \renewcommand*{\glossaryheader}{}%

```

No group headings (if heading is required, add `\glsinlinedopostchild` to start definition in case heading follows a child entry):

```

7377 \renewcommand*{\glsgroupheading}[1]{}%

```

Just display separator followed by name and description:

```

7378 \renewcommand{\glossentry}[2]{%
7379 \glsinlinedopostchild
7380 \gls@inlinesep

```

```

7381 \glsentryitem{##1}%
7382 \glsinlinenameformat{##1}{%
7383 \glossentryname{##1}%
7384 }%
7385 \ifglstdescsuppressed{##1}%
7386 {%
7387 \glsinlineemptydescformat
7388 {%
7389 \glossentrysymbol{##1}%
7390 }%
7391 {%
7392 ##2%
7393 }%
7394 }%
7395 {%
7396 \ifglshasdesc{##1}%
7397 {\glsinlinedescformat{\glossentrydesc{##1}}{\glossentrysymbol{##1}}{##2}}%
7398 {\glsinlineemptydescformat{\glossentrysymbol{##1}}{##2}}%
7399 }%
7400 \ifglshaschildren{##1}%
7401 {%
7402 \glsresetsubentrycounter
7403 \glsinlineparentchildseparator
7404 \def\gls@inlinesubsep{}%
7405 \def\gls@inlinepostchild{\glsinlinepostchild}%
7406 }%
7407 {}%
7408 \def\gls@inlinesep{\glsinlineseparator}%
7409 }%

```

Sub-entries display description:

```

7410 \renewcommand{\subglossentry}[3]{%
7411 \gls@inlinesubsep%
7412 \glsinlinesubnameformat{##2}{%
7413 \glossentryname{##2}}%
7414 \glssubentryitem{##2}%
7415 \glsinlinesubdescformat{\glossentrydesc{##2}}{\glossentrysymbol{##2}}{##3}%
7416 \def\gls@inlinesubsep{\glsinlinesubseparator}%
7417 }%

```

Nothing special between groups:

```

7418 \renewcommand*{\glsgroupskip}{}%
7419 }

```

`\glsinlinedopostchild`

```

7420 \newcommand*{\glsinlinedopostchild}{%
7421 \gls@inlinepostchild
7422 \def\gls@inlinepostchild{}%
7423 }

```

`\glsinlineseparator` Separator to use between entries.  
7424 `\newcommand*\glsinlineseparator}{;\space}`

`\glsinlinesubseparator` Separator to use between sub-entries.  
7425 `\newcommand*\glsinlinesubseparator}{,\space}`

`\glsinlineparentchildseparator` Separator to use between parent and children.  
7426 `\newcommand*\glsinlineparentchildseparator}{:\space}`

`\glsinlinepostchild` Hook to use between child and next entry  
7427 `\newcommand*\glsinlinepostchild}{}`

`\glspostinline` Terminator for inline glossary.  
7428 `\newcommand*\glspostinline}{\glspostdescription\space}`

`\glsinlinenameformat` Formats the name of the entry (first argument label, second argument name):  
7429 `\newcommand*\glsinlinenameformat}[2]{\glstarget{#1}{#2}}`

`\glsinlinedescformat` Formats the entry's description, symbol and location list:  
7430 `\newcommand*\glsinlinedescformat}[3]{\space#1}`

`\glsinlineemptydescformat` Formats the entry's symbol and location list when the description is empty:  
7431 `\newcommand*\glsinlineemptydescformat}[2]{}`

`\glsinlinesubnameformat` Formats the name of the subentry (first argument label, second argument name):  
7432 `\newcommand*\glsinlinesubnameformat}[2]{\glstarget{#1}{}}`

`\glsinlinesubdescformat` Formats the subentry's description, symbol and location list:  
7433 `\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.

7434 `\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.

7435 `\newglossarystyle{list}{%`



Use description environment:

```
7436 \renewenvironment{theglossary}%  
7437 {\begin{description}}{\end{description}}%
```

No header at the start of the environment:

```
7438 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7439 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries start a new item in the list:

```
7440 \renewcommand*{\glossentry}[2]{%  
7441 \item[\glsentryitem{##1}]%  
7442 \glstarget{##1}{\glossentryname{##1}}]  
7443 \glossentrydesc{##1}\glspostdescription\space ##2}%
```

Sub-entries continue on the same line:

```
7444 \renewcommand*{\subglossentry}[3]{%  
7445 \glssubentryitem{##2}%  
7446 \glstarget{##2}{\strut}%  
7447 \glossentrydesc{##2}\glspostdescription\space ##3.}%  
7448% \end{macrocode}  
7449% Add vertical space between groups:  
7450%\changes{3.03}{2012/09/21}{added check for glsnogroupskip}  
7451% \begin{macrocode}  
7452 \renewcommand*{\glsgroupskip}{\ifglsgroupskip\else\indexspace\fi}%  
7453 }
```

**listgroup** The listgroup style is like the list style, but the glossary groups have headings.

```
7454 \newglossarystyle{listgroup}{%
```

Base it on the list style:

```
7455 \setglossarystyle{list}%
```

Each group has a heading:

```
7456 \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.

```
7457 \newglossarystyle{listhypergroup}{%
```

Base it on the list style:

```
7458 \setglossarystyle{list}%
```

Add navigation links at the start of the environment:

```
7459 \renewcommand*{\glossaryheader}{%  
7460 \item[\glsnavigation]}%
```

Each group has a heading with a hypertarget:

```
7461 \renewcommand*{\glsgroupheading}[1]{%  
7462 \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.

```
7463 \newglossarystyle{altlist}{%
```

Base it on the list style:

```
7464 \setglossarystyle{list}%
```

Main (level 0) entries start a new item in the list with a line break after the entry name:

```
7465 \renewcommand*{\glossentry}[2]{%
7466 \item[\glsentryitem{##1}%
7467 \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.

```
7468 \mbox{} \par \nobreak \afterheading
7469 \glossentrydesc{##1} \glspostdescription \space ##2}%
```

Sub-entries start a new paragraph:

```
7470 \renewcommand{\subglossentry}[3]{%
7471 \par
7472 \glssubentryitem{##2}%
7473 \glstarget{##2}{\strut} \glossentrydesc{##2} \glspostdescription \space ##3}%
7474 }
```

**altlistgroup** The altlistgroup glossary style is like the altlist style, but the glossary groups have headings.

```
7475 \newglossarystyle{altlistgroup}{%
```

Base it on the altlist style:

```
7476 \setglossarystyle{altlist}%
```

Each group has a heading:

```
7477 \renewcommand*{\glsgroupheading}[1]{\item[\glsgrouptitle{##1}]}
```

**altlisthypergroup** The altlisthypergroup glossary style is like the altlistgroup style, but has a set of links to the groups at the start of the glossary.

```
7478 \newglossarystyle{altlisthypergroup}{%
```

Base it on the altlist style:

```
7479 \setglossarystyle{altlist}%
```

Add navigation links at the start of the environment:

```
7480 \renewcommand*{\glossaryheader}{%
7481 \item[\glsnavigation]}%
```

Each group has a heading with a hypertarget:

```
7482 \renewcommand*{\glsgroupheading}[1]{%
7483 \item[\glsnavhypertarget{##1}{\glsgrouptitle{##1}}]}
```

`listdotted` The `listdotted` glossary style was supplied by Axel Menzel. I've modified it slightly so that the distance from the start of the name to the end of the dotted line is specified by `\glslistdottedwidth`. Note that this style ignores the page numbers as well as the symbol. Sub-entries are displayed in the same way as top-level entries.

```
7484 \newglossarystyle{listdotted}{%
```

Base it on the list style:

```
7485 \setglossarystyle{list}{%
```

Each main (level 0) entry starts a new item:

```
7486 \renewcommand*{\glossentry}[2]{%
7487 \item[]\makebox[\glslistdottedwidth][l]{%
7488 \glentryitem{##1}%
7489 \glstarget{##1}{\glossentryname{##1}}%
7490 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}\glossentrydesc{##1}}%
```

Sub entries have the same format as main entries:

```
7491 \renewcommand*{\subglossentry}[3]{%
7492 \item[]\makebox[\glslistdottedwidth][l]{%
7493 \glssubentryitem{##2}%
7494 \glstarget{##2}{\glossentryname{##2}}%
7495 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}\glossentrydesc{##2}}%
7496 }
```

`\glslistdottedwidth`

```
7497 \newlength\glslistdottedwidth
7498 \setlength{\glslistdottedwidth}{.5\hsize}
```

`sublistdotted` This style is similar to the `glostylelistdotted` style, except that the main entries just have the name displayed.

```
7499 \newglossarystyle{sublistdotted}{%
```

Base it on the `listdotted` style:

```
7500 \setglossarystyle{listdotted}{%
```

Main (level 0) entries just display the name:

```
7501 \renewcommand*{\glossentry}[2]{%
7502 \item[\glentryitem{##1}\glstarget{##1}{\glossentryname{##1}}}%
7503 }
```

## 5.4 Glossary Styles using `longtable` (the `glossary-long` package)

The glossary styles defined in the package used the `longtable` environment in the glossary.

```
7504 \ProvidesPackage{glossary-long}[2013/11/14 v4.0 (NLCT)]
```

Requires the package:

```
7505 \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`.)

```
7506 \ifundefined{glsdescwidth}{%
7507   \newlength{glsdescwidth
7508   \setlength{glsdescwidth}{0.6\hsize}
7509 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column.

```
7510 \ifundefined{glspagelistwidth}{%
7511   \newlength{glspagelistwidth
7512   \setlength{glspagelistwidth}{0.1\hsize}
7513 }{}
```

`long` The long glossary style command which uses the `longtable` environment:

```
7514 \newglossarystyle{long}{%
```

Use `longtable` with two columns:

```
7515   \renewenvironment{theglossary}%
7516     {\begin{longtable}{lp{glsdescwidth}}}%
7517     {\end{longtable}}%
```

Do nothing at the start of the environment:

```
7518   \renewcommand*{\glossaryheader}{}%
```

No heading between groups:

```
7519   \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries displayed in a row:

```
7520   \renewcommand{\glossentry}[2]{%
7521     \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7522     \glossentrydesc{##1}\glspostdescription\space ##2\tabularnewline
7523   }%
```

Sub entries displayed on the following row without the name:

```
7524   \renewcommand{\subglossentry}[3]{%
7525     &
7526     \glssubentryitem{##2}%
7527     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space
7528     ##3\tabularnewline
7529   }%
```

Blank row between groups:

```
7530   \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else &
7531   \tabularnewline\fi}%
7532 }
```

`longborder` The `longborder` style is like the above, but with horizontal and vertical lines:

```
7533 \newglossarystyle{longborder}{%
```

Base it on the `glostylelong` style:

```
7534 \setglossarystyle{long}%
```

Use `longtable` with two columns with vertical lines between each column:

```
7535 \renewenvironment{theglossary}{%  
7536 \begin{longtable}{|l|p{\glstdescwidth}|}{\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
7537 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%  
7538 }
```

`longheader` The `longheader` style is like the `long` style but with a header:

```
7539 \newglossarystyle{longheader}{%
```

Base it on the `glostylelong` style:

```
7540 \setglossarystyle{long}%
```

Set the table's header:

```
7541 \renewcommand*{\glossaryheader}{%  
7542 \bfseries \entryname & \bfseries \descriptionname\tabularnewline\endhead}%  
7543 }
```

`longheaderborder` The `longheaderborder` style is like the `long` style but with a header and border:

```
7544 \newglossarystyle{longheaderborder}{%
```

Base it on the `glostylelongborder` style:

```
7545 \setglossarystyle{longborder}%
```

Set the table's header and add horizontal line to table's foot:

```
7546 \renewcommand*{\glossaryheader}{%  
7547 \hline\bfseries \entryname & \bfseries  
7548 \descriptionname\tabularnewline\hline  
7549 \endhead  
7550 \hline\endfoot}%  
7551 }
```

`long3col` The `long3col` style is like `long` but with 3 columns

```
7552 \newglossarystyle{long3col}{%
```

Use a `longtable` with 3 columns:

```
7553 \renewenvironment{theglossary}%  
7554 {\begin{longtable}{lp{\glstdescwidth}p{\glspagelistwidth}}}%  
7555 {\end{longtable}}%
```

No table header:

```
7556 \renewcommand*{\glossaryheader}{}%
```

No headings between groups:

```
7557 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
7558 \renewcommand{\glossentry}[2]{%
7559   \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7560   \glossentrydesc{##1} & ##2\tabularnewline
7561 }%
```

Sub-entries on a separate row (no name, description in second column, page list in third column):

```
7562 \renewcommand{\subglossentry}[3]{%
7563   &
7564   \glssubentryitem{##2}%
7565   \glstarget{##2}{\strut}\glossentrydesc{##2} &
7566   ##3\tabularnewline
7567 }%
```

Blank row between groups:

```
7568 \renewcommand*{\glsgroupskip}{%
7569   \ifglsgnogroupskip\else & &\tabularnewline\fi}%
7570 }
```

**long3colborder** The long3colborder style is like the long3col style but with a border:

```
7571 \newglossarystyle{long3colborder}{%
```

Base it on the glostylelong3col style:

```
7572 \setglossarystyle{long3col}%
```

Use a longtable with 3 columns with vertical lines around them:

```
7573 \renewenvironment{theglossary}%
7574   {\begin{longtable}{|l|p{\glsgdescwidth}|p{\glspagelistwidth}|}%
7575   {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
7576 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
7577 }
```

**long3colheader** The long3colheader style is like long3col but with a header row:

```
7578 \newglossarystyle{long3colheader}{%
```

Base it on the glostylelong3col style:

```
7579 \setglossarystyle{long3col}%
```

Set the table's header:

```
7580 \renewcommand*{\glossaryheader}{%
7581   \bfseries\entryname&\bfseries\descriptionname&
7582   \bfseries\pagelistname\tabularnewline\endhead}%
7583 }
```

**long3colheaderborder** The long3colheaderborder style is like the above but with a border

```
7584 \newglossarystyle{long3colheaderborder}{%
```

Base it on the `glostylelong3colborder` style:

```
7585 \setglossarystyle{long3colborder}{%
```

Set the table's header and add horizontal line at table's foot:

```
7586 \renewcommand*{\glossaryheader}{{%  
7587 \hline  
7588 \bfseries\entryname&\bfseries\descriptionname&  
7589 \bfseries\pagelistname\tabularnewline\hline\endhead  
7590 \hline\endfoot}%  
7591 }
```

`long4col` The `long4col` style has four columns where the third column contains the value of the associated symbol key.

```
7592 \newglossarystyle{long4col}{%
```

Use a `longtable` with 4 columns:

```
7593 \renewenvironment{theglossary}{%  
7594 {\begin{longtable}{llll}}%  
7595 {\end{longtable}}%
```

No table header:

```
7596 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7597 \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):

```
7598 \renewcommand{\glossentry}[2]{%  
7599 \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &  
7600 \glossentrydesc{##1} &  
7601 \glossentrysymbol{##1} &  
7602 ##2\tabularnewline  
7603 }%
```

Sub entries on a single row with no name (description in second column, symbol in third column, page list in last column):

```
7604 \renewcommand{\subglossentry}[3]{%  
7605 &  
7606 \glssubentryitem{##2}%  
7607 \glstarget{##2}{\strut}\glossentrydesc{##2} &  
7608 \glossentrysymbol{##2} & ##3\tabularnewline  
7609 }%
```

Blank row between groups:

```
7610 \renewcommand*{\glsgroupskip}{%  
7611 \ifglsgnোগroupskip\else & & \tabularnewline\fi}%  
7612 }
```

`long4colheader` The `long4colheader` style is like `long4col` but with a header row.

```
7613 \newglossarystyle{long4colheader}{%
```

Base it on the `glostylelong4col` style:

```
7614 \setglossarystyle{long4col}%
```

Table has a header:

```
7615 \renewcommand*{\glossaryheader}{%  
7616 \bfseries\entryname&\bfseries\descriptionname&  
7617 \bfseries \symbolname&  
7618 \bfseries\pagelistname\tabularnewline\endhead}%  
7619 }
```

`long4colborder` The `long4colborder` style is like `long4col` but with a border.

```
7620 \newglossarystyle{long4colborder}{%
```

Base it on the `glostylelong4col` style:

```
7621 \setglossarystyle{long4col}%
```

Use a longtable with 4 columns surrounded by vertical lines:

```
7622 \renewenvironment{theglossary}%  
7623 {\begin{longtable}{|l|l|l|l|}}%  
7624 {\end{longtable}}%
```

Add horizontal lines to the head and foot of the table:

```
7625 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%  
7626 }
```

`long4colheaderborder` The `long4colheaderborder` style is like the above but with a border.

```
7627 \newglossarystyle{long4colheaderborder}{%
```

Base it on the `glostylelong4col` style:

```
7628 \setglossarystyle{long4col}%
```

Use a longtable with 4 columns surrounded by vertical lines:

```
7629 \renewenvironment{theglossary}%  
7630 {\begin{longtable}{|l|l|l|l|}}%  
7631 {\end{longtable}}%
```

Add table header and horizontal line at the table's foot:

```
7632 \renewcommand*{\glossaryheader}{%  
7633 \hline\bfseries\entryname&\bfseries\descriptionname&  
7634 \bfseries \symbolname&  
7635 \bfseries\pagelistname\tabularnewline\hline\endhead  
7636 \hline\endfoot}%  
7637 }
```

`altlong4col` The `altlong4col` style is like the `long4col` style but can have multiline descriptions and page lists.

```
7638 \newglossarystyle{altlong4col}{%
```

Base it on the `glostylelong4col` style:

```
7639 \setglossarystyle{long4col}%
```



Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
7640 \renewenvironment{theglossary}%
7641   {\begin{longtable}{lp{\glsdescwidth}lp{\glspagelistwidth}}}%
7642   {\end{longtable}}%
7643 }
```

**altlong4colheader** The altlong4colheader style is like altlong4col but with a header row.

```
7644 \newglossarystyle{altlong4colheader}{%
```

Base it on the glostylelong4colheader style:

```
7645 \setglossarystyle{long4colheader}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
7646 \renewenvironment{theglossary}%
7647   {\begin{longtable}{lp{\glsdescwidth}lp{\glspagelistwidth}}}%
7648   {\end{longtable}}%
7649 }
```

**altlong4colborder** The altlong4colborder style is like altlong4col but with a border.

```
7650 \newglossarystyle{altlong4colborder}{%
```

Base it on the glostylelong4colborder style:

```
7651 \setglossarystyle{long4colborder}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
7652 \renewenvironment{theglossary}%
7653   {\begin{longtable}{|lp{\glsdescwidth}|lp{\glspagelistwidth}|}}%
7654   {\end{longtable}}%
7655 }
```

**ong4colheaderborder** The altlong4colheaderborder style is like the above but with a header as well as a border.

```
7656 \newglossarystyle{altlong4colheaderborder}{%
```

Base it on the glostylelong4colheaderborder style:

```
7657 \setglossarystyle{long4colheaderborder}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
7658 \renewenvironment{theglossary}%
7659   {\begin{longtable}{|lp{\glsdescwidth}|lp{\glspagelistwidth}|}}%
7660   {\end{longtable}}%
7661 }
```

## 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.

7662 \ProvidesPackage{glossary-longragged}[2014/07/30 v4.08 (NLCT)]

Requires the package:

7663 \RequirePackage{array}

Requires the package:

7664 \RequirePackage{longtable}

`\glsdescwidth` This is a length that governs the width of the description column. This may have already been defined.

```
7665 \@ifundefined{glsdescwidth}{%
7666   \newlength{glsdescwidth
7667   \setlength{glsdescwidth}{0.6\hsize}
7668 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

```
7669 \@ifundefined{glspagelistwidth}{%
7670   \newlength{glspagelistwidth
7671   \setlength{glspagelistwidth}{0.1\hsize}
7672 }{}
```

`longragged` The longragged glossary style is like the long but uses ragged right formatting for the description column.

7673 \newglossarystyle{longragged}{%

Use longtable with two columns:

```
7674   \renewenvironment{theglossary}{%
7675     {\begin{longtable}{l>{\raggedright}p{glsdescwidth}}}%
7676     {\end{longtable}}%
```

Do nothing at the start of the environment:

7677 \renewcommand\*{\glossaryheader}{}%

No heading between groups:

7678 \renewcommand\*{\glsgroupheading}[1]{}%

Main (level 0) entries displayed in a row:

```
7679 \renewcommand{\glossentry}[2]{%
7680   \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7681   \glossentrydesc{##1}\glspostdescription\space ##2%
7682   \tabularnewline
7683 }%
```

Sub entries displayed on the following row without the name:

```

7684 \renewcommand{\subglossentry}[3]{%
7685     &
7686     \glssubentryitem{##2}%
7687     \glstarget{##2}{\strut}\glossentrydesc{##2}%
7688     \glspostdescription\space ##3%
7689     \tabularnewline
7690 }%
```

Blank row between groups:

```

7691 \renewcommand*{\glsgroupskip}{\ifglsgnোগroupskip\else & \tabularnewline\fi}%
7692 }
```

**longraggedborder** The longraggedborder style is like the above, but with horizontal and vertical lines:

```

7693 \newglossarystyle{longraggedborder}{%
```

Base it on the glostylelongragged style:

```

7694 \setglossarystyle{longragged}%
```

Use longtable with two columns with vertical lines between each column:

```

7695 \renewenvironment{theglossary}{%
7696     \begin{longtable}{|l|>{\raggedright}p{\glstdescwidth}|}%
7697     {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```

7698 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
7699 }
```

**longraggedheader** The longraggedheader style is like the longragged style but with a header:

```

7700 \newglossarystyle{longraggedheader}{%
```

Base it on the glostylelongragged style:

```

7701 \setglossarystyle{longragged}%
```

Set the table's header:

```

7702 \renewcommand*{\glossaryheader}{%
7703     \bfseries \entryname & \bfseries \descriptionname
7704     \tabularnewline\endhead}%
7705 }
```

**longraggedheaderborder** The longraggedheaderborder style is like the longragged style but with a header and border:

```

7706 \newglossarystyle{longraggedheaderborder}{%
```

Base it on the glostylelongraggedborder style:

```

7707 \setglossarystyle{longraggedborder}%
```

Set the table's header and add horizontal line to table's foot:

```

7708 \renewcommand*{\glossaryheader}{%
7709     \hline\bfseries \entryname & \bfseries \descriptionname
```

```

7710 \tabularnewline\hline
7711 \endhead
7712 \hline\endfoot}%
7713 }

```

`longragged3col` The `longragged3col` style is like `longragged` but with 3 columns

```

7714 \newglossarystyle{longragged3col}{%

```

Use a longtable with 3 columns:

```

7715 \renewenvironment{theglossary}%
7716 {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}%
7717 >{\raggedright}p{\glspagelistwidth}}}%
7718 {\end{longtable}}%

```

No table header:

```

7719 \renewcommand*{\glossaryheader}{}%

```

No headings between groups:

```

7720 \renewcommand*{\glsgroupheading}[1]{}%

```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```

7721 \renewcommand{\glossentry}[2]{%
7722 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7723 \glossentrydesc{##1} & ##2\tabularnewline
7724 }%

```

Sub-entries on a separate row (no name, description in second column, page list in third column):

```

7725 \renewcommand{\subglossentry}[3]{%
7726 &
7727 \glssubentryitem{##2}%
7728 \glstarget{##2}{\strut}\glossentrydesc{##2} &
7729 ##3\tabularnewline
7730 }%

```

Blank row between groups:

```

7731 \renewcommand*{\glsgroupskip}{%
7732 \ifglsnogroupskip\else & &\tabularnewline\fi}%
7733 }

```

`longragged3colborder` The `longragged3colborder` style is like the `longragged3col` style but with a border:

```

7734 \newglossarystyle{longragged3colborder}{%

```

Base it on the `glostylelongragged3col` style:

```

7735 \setglossarystyle{longragged3col}%

```

Use a longtable with 3 columns with vertical lines around them:

```

7736 \renewenvironment{theglossary}%
7737 {\begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|}%
7738 >{\raggedright}p{\glspagelistwidth}|}%
7739 {\end{longtable}}%

```

Place horizontal lines at the head and foot of the table:

```
7740 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
7741 }
```

`longragged3colheader` The `longragged3colheader` style is like `longragged3col` but with a header row:

```
7742 \newglossarystyle{longragged3colheader}{%
```

Base it on the `glostylelongragged3col` style:

```
7743 \setglossarystyle{longragged3col}%
```

Set the table's header:

```
7744 \renewcommand*{\glossaryheader}{%
7745 \bfseries\entryname&\bfseries\descriptionname&
7746 \bfseries\pagelistname\tabularnewline\endhead}%
7747 }
```

`longragged3colheaderborder` The `longragged3colheaderborder` style is like the above but with a border

```
7748 \newglossarystyle{longragged3colheaderborder}{%
```

Base it on the `glostylelongragged3colborder` style:

```
7749 \setglossarystyle{longragged3colborder}%
```

Set the table's header and add horizontal line at table's foot:

```
7750 \renewcommand*{\glossaryheader}{%
7751 \hline
7752 \bfseries\entryname&\bfseries\descriptionname&
7753 \bfseries\pagelistname\tabularnewline\hline\endhead
7754 \hline\endfoot}%
7755 }
```

`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.

```
7756 \newglossarystyle{altlongragged4col}{%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
7757 \renewenvironment{theglossary}%
7758 {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%
7759 >{\raggedright}p{\glspagelistwidth}}}%
7760 {\end{longtable}}%
```

No table header:

```
7761 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7762 \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):

```
7763 \renewcommand{\glossentry}[2]{}%
```

```

7764 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7765 \glossentrydesc{##1} & \glossentrysymbol{##1} &
7766 ##2\tabularnewline
7767 }%

```

Sub entries on a single row with no name (description in second column, symbol in third column, page list in last column):

```

7768 \renewcommand{\subglossentry}[3]{%
7769 &
7770 \glssubentryitem{##2}%
7771 \glstarget{##2}{\strut}\glossentrydesc{##2} &
7772 \glossentrysymbol{##2} & ##3\tabularnewline
7773 }%

```

Blank row between groups:

```

7774 \renewcommand*\glsgroupskip{%
7775 \ifglisnogroupskip\else & & \tabularnewline\fi}%
7776 }

```

**longragged4colheader** The `altlongragged4colheader` style is like `altlongragged4col` but with a header row.

```

7777 \newglossarystyle{altlongragged4colheader}{%

```

Base it on the `glostylealtlongragged4col` style:

```

7778 \setglossarystyle{altlongragged4col}%

```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```

7779 \renewenvironment{theglossary}%
7780 {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%
7781 >{\raggedright}p{\glspagelistwidth}}}%
7782 {\end{longtable}}%

```

Table has a header:

```

7783 \renewcommand*\glossaryheader{%
7784 \bfseries\entryname&\bfseries\descriptionname&
7785 \bfseries \symbolname&
7786 \bfseries\pagelistname\tabularnewline\endhead}%
7787 }

```

**longragged4colborder** The `altlongragged4colborder` style is like `altlongragged4col` but with a border.

```

7788 \newglossarystyle{altlongragged4colborder}{%

```

Base it on the `glostylealtlongragged4col` style:

```

7789 \setglossarystyle{altlongragged4col}%

```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```

7790 \renewenvironment{theglossary}%
7791 {\begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|l|%
7792 >{\raggedright}p{\glspagelistwidth}|}%
7793 {\end{longtable}}%

```

Add horizontal lines to the head and foot of the table:

```
7794 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
7795 }
```

`ged4colheaderborder` The `altlongragged4colheaderborder` style is like the above but with a header as well as a border.

```
7796 \newglossarystyle{altlongragged4colheaderborder}{%
```

Base it on the `glostylealtlongragged4col` style:

```
7797 \setglossarystyle{altlongragged4col}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
7798 \renewenvironment{theglossary}%
7799 {\begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|l|}%
7800 >{\raggedright}p{\glspagelistwidth}|}%
7801 {\end{longtable}}%
```

Add table header and horizontal line at the table's foot:

```
7802 \renewcommand*{\glossaryheader}{%
7803 \hline\bfseries\entryname&\bfseries\descriptionname&
7804 \bfseries \symbolname&
7805 \bfseries\pagelistname\tabularnewline\hline\endhead
7806 \hline\endfoot}%
7807 }
```

## 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.

```
7808 \ProvidesPackage{glossary-mcols}[2013/11/14 v4.0 (NLCT)]
```

Required packages:

```
7809 \RequirePackage{multicol}
7810 \RequirePackage{glossary-tree}
```

`\glscols` Define macro in which to store the number of columns. (Defaults to 2.)

```
7811 \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.)

```
7812 \newglossarystyle{mcolindex}{%
7813 \setglossarystyle{index}%
7814 \renewenvironment{theglossary}%
7815 {%
```

```

7816 \begin{multicols}{\glsmcols}
7817 \setlength{\parindent}{0pt}%
7818 \setlength{\parskip}{0pt plus 0.3pt}%
7819 \let\item\@idxitem}%
7820 {\end{multicols}}%
7821 }

```

**mcolindexgroup** As mcolindex but has headings:

```

7822 \newglossarystyle{mcolindexgroup}{%
7823 \setglossarystyle{mcolindex}%
7824 \renewcommand*{\glsgroupheading}[1]{%
7825 \item\textbf{\glsgroupheading{##1}}\indexspace}%
7826 }

```

**mcolindexhypergroup** The mcolindexhypergroup style is like the mcolindexgroup style but has hyper navigation.

```

7827 \newglossarystyle{mcolindexhypergroup}{%
  Base it on the glostylemcolindex style:
7828 \setglossarystyle{mcolindex}%
  Put navigation links to the groups at the start of the glossary:
7829 \renewcommand*{\glossaryheader}{%
7830 \item\textbf{\glshypernavigation}\indexspace}%
  Add a heading for each group (with a target). The group's title is in bold followed
  by a vertical gap.
7831 \renewcommand*{\glsgroupheading}[1]{%
7832 \item\textbf{\glshypernavhypertarget{##1}}{\glsgroupheading{##1}}}%
7833 \indexspace}%
7834 }

```

**mcoltree** Multi-column index style. Same as the tree, but puts the glossary in multiple columns.

```

7835 \newglossarystyle{mcoltree}{%
7836 \setglossarystyle{tree}%
7837 \renewenvironment{theglossary}%
7838 {%
7839 \begin{multicols}{\glsmcols}
7840 \setlength{\parindent}{0pt}%
7841 \setlength{\parskip}{0pt plus 0.3pt}%
7842 }%
7843 {\end{multicols}}%
7844 }

```

**mcoltreegroup** Like the mcoltree style but the glossary groups have headings.

```

7845 \newglossarystyle{mcoltreegroup}{%
  Base it on the glostylemcoltree style:
7846 \setglossarystyle{mcoltree}%

```



Each group has a heading (in bold) followed by a vertical gap):

```
7847 \renewcommand{\glsgroupheading}[1]{\par
7848 \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
7849 }
```

**mcoltreehypergroup** The mcoltreehypergroup style is like the treegroup style, but has a set of links to the groups at the start of the glossary.

```
7850 \newglossarystyle{mcoltreehypergroup}{%
```

Base it on the glostylemcoltree style:

```
7851 \setglossarystyle{mcoltree}%
```

Put navigation links to the groups at the start of the theglossary environment:

```
7852 \renewcommand*{\glossaryheader}{%
7853 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
7854 \renewcommand*{\glsgroupheading}[1]{%
7855 \par\noindent
7856 \textbf{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}\par
7857 \indexspace}%
7858 }
```

**mcoltreenoname** Multi-column index style. Same as the treenoname, but puts the glossary in multiple columns.

```
7859 \newglossarystyle{mcoltreenoname}{%
7860 \setglossarystyle{treenoname}%
7861 \renewenvironment{theglossary}%
7862 {%
7863 \begin{multicols}{\glsmcols}
7864 \setlength{\parindent}{0pt}%
7865 \setlength{\parskip}{0pt plus 0.3pt}%
7866 }%
7867 {\end{multicols}}}%
7868 }
```

**mcoltreenonamegroup** Like the mcoltreenoname style but the glossary groups have headings.

```
7869 \newglossarystyle{mcoltreenonamegroup}{%
```

Base it on the glostylemcoltreenoname style:

```
7870 \setglossarystyle{mcoltreenoname}%
```

Give each group a heading:

```
7871 \renewcommand{\glsgroupheading}[1]{\par
7872 \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
7873 }
```

**treenonamehypergroup** The mcoltreenonamehypergroup style is like the mcoltreenonamegroup style, but has a set of links to the groups at the start of the glossary.

```
7874 \newglossarystyle{mcoltreenonamehypergroup}{%
```

Base it on the `glostylemcoltreenoname` style:

```
7875 \setglossarystyle{mcoltreenoname}%
```

Put navigation links to the groups at the start of the `theglossary` environment:

```
7876 \renewcommand*{\glossaryheader}{%
```

```
7877 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
7878 \renewcommand*{\glsgroupheading}[1]{%
```

```
7879 \par\noindent
```

```
7880 \textbf{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}\par
```

```
7881 \indexspace}%
```

```
7882 }
```

`mcolalmtree` Multi-column index style. Same as the `almtree`, but puts the glossary in multiple columns.

```
7883 \newglossarystyle{mcolalmtree}{%
```

```
7884 \setglossarystyle{almtree}%
```

```
7885 \renewenvironment{theglossary}%
```

```
7886 {%
```

```
7887 \begin{multicols}{\glsmcols}
```

```
7888 \def\@gls@prevlevel{-1}%
```

```
7889 \mbox{}\par
```

```
7890 }%
```

```
7891 {\par\end{multicols}}%
```

```
7892 }
```

`mcolalmtreegroup` Like the `mcolalmtree` style but the glossary groups have headings.

```
7893 \newglossarystyle{mcolalmtreegroup}{%
```

Base it on the `glostylemcolalmtree` style:

```
7894 \setglossarystyle{mcolalmtree}%
```

Give each group a heading.

```
7895 \renewcommand{\glsgroupheading}[1]{\par
```

```
7896 \def\@gls@prevlevel{-1}%
```

```
7897 \hangindent0pt\relax
```

```
7898 \parindent0pt\relax
```

```
7899 \textbf{\glsgrouptitle{##1}}\par\indexspace}%
```

```
7900 }
```

`olalmtreehypergroup` The `mcolalmtreehypergroup` style is like the `mcolalmtreegroup` style, but has a set of links to the groups at the start of the glossary.

```
7901 \newglossarystyle{mcolalmtreehypergroup}{%
```

Base it on the `glostylemcolalmtree` style:

```
7902 \setglossarystyle{mcolalmtree}%
```

Put the navigation links in the header

```
7903 \renewcommand*{\glossaryheader}{%
7904   \par
7905   \def\@gls@prevlevel{-1}%
7906   \hangindent0pt\relax
7907   \parindent0pt\relax
7908   \textbf{\glsnavigation}\par\indexspace}%
```

Put a hypertarget at the start of each group

```
7909 \renewcommand*{\glsgroupheading}[1]{%
7910   \par
7911   \def\@gls@prevlevel{-1}%
7912   \hangindent0pt\relax
7913   \parindent0pt\relax
7914   \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
7915   \indexspace}}
```

## 5.7 Glossary Styles using supertabular environment (glossary-super package)

The glossary styles defined in the package use the supertabular environment.

```
7916 \ProvidesPackage{glossary-super}[2013/11/14 v4.0 (NLCT)]
```

Requires the package:

```
7917 \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.

```
7918 \@ifundefined{glsdescwidth}{%
7919   \newlength\glsdescwidth
7920   \setlength{\glsdescwidth}{0.6\hsize}
7921 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined if has been loaded.

```
7922 \@ifundefined{glspagelistwidth}{%
7923   \newlength\glspagelistwidth
7924   \setlength{\glspagelistwidth}{0.1\hsize}
7925 }{}
```

`super` The super glossary style uses the supertabular environment (it uses lengths defined in the package.)

```
7926 \newglossarystyle{super}{%
```

Put the glossary in a supertabular environment with two columns and no head or tail:

```
7927   \renewenvironment{theglossary}{%
7928     {\tablehead{}\tabletail}%
7929     \begin{supertabular}{lp{\glsdescwidth}}%
7930     {\end{supertabular}}%
```

Do nothing at the start of the table:

```
7931 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7932 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries put in a row (name in first column, description and page list in second column):

```
7933 \renewcommand{\glossentry}[2]{%
7934   \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7935   \glossentrydesc{##1}\glspostdescription\space ##2\tabularnewline
7936 }
```

Sub entries put in a row (no name, description and page list in second column):

```
7937 \renewcommand{\subglossentry}[3]{%
7938   &
7939   \glssubentryitem{##2}%
7940   \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space
7941   ##3\tabularnewline
7942 }
```

Blank row between groups:

```
7943 \renewcommand*{\glsgroupskip}{}%
7944 \ifglsnogroupskip\else & \tabularnewline\fi}%
7945 }
```

**superborder** The superborder style is like the above, but with horizontal and vertical lines:

```
7946 \newglossarystyle{superborder}{}%
```

Base it on the `glostylesuper` style:

```
7947 \setglossarystyle{super}{}%
```

Put the glossary in a `supertabular` environment with two columns and a horizontal line in the head and tail:

```
7948 \renewenvironment{theglossary}%
7949   {\tablehead{\hline}\tabletail{\hline}}%
7950   \begin{supertabular}{|l|p{\glsdescwidth}|}%
7951   {\end{supertabular}}%
7952 }
```

**superheader** The superheader style is like the super style, but with a header:

```
7953 \newglossarystyle{superheader}{}%
```

Base it on the `glostylesuper` style:

```
7954 \setglossarystyle{super}{}%
```

Put the glossary in a `supertabular` environment with two columns, a header and no tail:

```
7955 \renewenvironment{theglossary}%
7956   {\tablehead{\bfseries \entryname &
```

```

7957 \bfseries\descriptionname\tabularnewline}%
7958 \tabletail{}}%
7959 \begin{supertabular}{lp{\glsgdescwidth}}}%
7960 {\end{supertabular}}}%
7961 }

```

**superheaderborder** The superheaderborder style is like the super style but with a header and border:

```
7962 \newglossarystyle{superheaderborder}{%
```

Base it on the glostylesuper style:

```
7963 \setglossarystyle{super}%
```

Put the glossary in a supertabular environment with two columns, a header and horizontal lines above and below the table:

```

7964 \renewenvironment{theglossary}%
7965 {\tablehead{\hline\bfseries \entryname &
7966 \bfseries \descriptionname\tabularnewline\hline}%
7967 \tabletail{\hline}
7968 \begin{supertabular}{|lp{\glsgdescwidth}|}}%
7969 {\end{supertabular}}}%
7970 }

```

**super3col** The super3col style is like the super style, but with 3 columns:

```
7971 \newglossarystyle{super3col}{%
```

Put the glossary in a supertabular environment with three columns and no head or tail:

```

7972 \renewenvironment{theglossary}%
7973 {\tablehead{} \tabletail{}}%
7974 \begin{supertabular}{lp{\glsgdescwidth}p{\glspagelistwidth}}}%
7975 {\end{supertabular}}}%

```

Do nothing at the start of the table:

```
7976 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7977 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```

7978 \renewcommand{\glossentry}[2]{%
7979 \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7980 \glossentrydesc{##1} & ##2\tabularnewline
7981 }%

```

Sub entries on a row (no name, description in second column, page list in last column):

```

7982 \renewcommand{\subglossentry}[3]{%
7983 &
7984 \glssubentryitem{##2}%
7985 \glstarget{##2}{\strut}\glossentrydesc{##2} &

```

7986       ##3\tabularnewline

7987   }%

Blank row between groups:

7988   \renewcommand\*{\glsgroupskip}{%

7989       \ifglsgroupskip\else & &\tabularnewline\fi}%

7990 }

super3colborder   The super3colborder style is like the super3col style, but with a border:

7991 \newglossarystyle{super3colborder}{%

Base it on the glostylesuper3col style:

7992   \setglossarystyle{super3col}%

Put the glossary in a supertabular environment with three columns and a horizontal line in the head and tail:

7993   \renewenvironment{theglossary}{%

7994       {\tablehead{\hline}\tabletail{\hline}}%

7995       \begin{supertabular}{|l|p{\glsgdescwidth}|p{\glspagelistwidth}|}%

7996       {\end{supertabular}}%

7997 }

super3colheader   The super3colheader style is like the super3col style but with a header row:

7998 \newglossarystyle{super3colheader}{%

Base it on the glostylesuper3col style:

7999   \setglossarystyle{super3col}%

Put the glossary in a supertabular environment with three columns, a header and no tail:

8000   \renewenvironment{theglossary}{%

8001       {\tablehead{\bfseries\entryname&\bfseries\descriptionname&

8002           \bfseries\pagelistname\tabularnewline}\tabletail{}}%

8003       \begin{supertabular}{|lp{\glsgdescwidth}p{\glspagelistwidth}}%

8004       {\end{supertabular}}%

8005 }

per3colheaderborder   The super3colheaderborder style is like the super3col style but with a header and border:

8006 \newglossarystyle{super3colheaderborder}{%

Base it on the glostylesuper3colborder style:

8007   \setglossarystyle{super3colborder}%

Put the glossary in a supertabular environment with three columns, a header with horizontal lines and a horizontal line in the tail:

8008   \renewenvironment{theglossary}{%

8009       {\tablehead{\hline

8010           \bfseries\entryname&\bfseries\descriptionname&

8011           \bfseries\pagelistname\tabularnewline\hline}%

8012       \tabletail{\hline}}%

```

8013     \begin{supertabular}{|l|p{\glstdescwidth}|p{\glspagelistwidth}|}%
8014     {\end{supertabular}}}%
8015 }

```

**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.

```
8016 \newglossarystyle{super4col}{%
```

Put the glossary in a **supertabular** environment with four columns and no head or tail:

```

8017   \renewenvironment{theglossary}%
8018   {\tablehead{}\tabletail}%
8019   \begin{supertabular}{|l|l|l|l|}%
8020   \end{supertabular}}%

```

Do nothing at the start of the table:

```
8021   \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
8022   \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:

```

8023   \renewcommand{\glossentry}[2]{%
8024     \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
8025     \glossentrydesc{##1} &
8026     \glossentrysymbol{##1} & ##3\tabularnewline
8027   }%

```

Sub entries on a row with no name, the description in the second column, symbol in third column and page list in last column:

```

8028   \renewcommand{\subglossentry}[3]{%
8029     &
8030     \glssubentryitem{##2}%
8031     \glstarget{##2}{\strut}\glossentrydesc{##2} &
8032     \glossentrysymbol{##2} & ##3\tabularnewline
8033   }%

```

Blank row between groups:

```

8034   \renewcommand*{\glsgroupskip}{%
8035     \ifglsnogroupskip\else & & \tabularnewline\fi}%
8036 }

```

**super4colheader** The **super4colheader** style is like the **super4col** but with a header row.

```
8037 \newglossarystyle{super4colheader}{%
```

Base it on the **glostylesuper4col** style:

```
8038   \setglossarystyle{super4col}%
```

Put the glossary in a **supertabular** environment with four columns, a header and no tail:

```

8039 \renewenvironment{theglossary}%
8040   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
8041     \bfseries\symbolname &
8042     \bfseries\pagelistname\tabularnewline}%
8043   \tabletail{}}%
8044   \begin{supertabular}{|l|l|l|l|}%
8045   {\end{supertabular}}}%
8046 }

```

**super4colborder** The super4colborder style is like the super4col but with a border.

```

8047 \newglossarystyle{super4colborder}{%
      Base it on the glostylesuper4col style:
8048   \setglossarystyle{super4col}%
      Put the glossary in a supertabular environment with four columns and a horizontal line in the head and tail:
8049   \renewenvironment{theglossary}%
8050   {\tablehead{\hline}\tabletail{\hline}%
8051     \begin{supertabular}{|l|l|l|l|}%
8052     {\end{supertabular}}}%
8053 }

```

**per4colheaderborder** The super4colheaderborder style is like the super4col but with a header and border.

```

8054 \newglossarystyle{super4colheaderborder}{%
      Base it on the glostylesuper4col style:
8055   \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:
8056   \renewenvironment{theglossary}%
8057   {\tablehead{\hline\bfseries\entryname&\bfseries\descriptionname&
8058     \bfseries\symbolname &
8059     \bfseries\pagelistname\tabularnewline\hline}%
8060   \tabletail{\hline}%
8061   \begin{supertabular}{|l|l|l|l|}%
8062   {\end{supertabular}}}%
8063 }

```

**altsuper4col** The altsuper4col glossary style is like super4col but has provision for multiline descriptions.

```

8064 \newglossarystyle{altsuper4col}{%
      Base it on the glostylesuper4col style:
8065   \setglossarystyle{super4col}%
      Put the glossary in a supertabular environment with four columns and no head or tail:

```



```

8066 \renewenvironment{theglossary}%
8067 {\tablehead{}\tabletail{}}%
8068 \begin{supertabular}{lp{\glsgdescwidth}lp{\glspagelistwidth}}}%
8069 {\end{supertabular}}}%
8070 }

```

**altsuper4colheader** The altsuper4colheader style is like the altsuper4col but with a header row.

```

8071 \newglossarystyle{altsuper4colheader}{%
      Base it on the glostylesuper4colheader style:
8072 \setglossarystyle{super4colheader}%
      Put the glossary in a supertabular environment with four columns, a header and
      no tail:
8073 \renewenvironment{theglossary}%
8074 {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
8075 \bfseries\symbolname &
8076 \bfseries\pagelistname\tabularnewline}\tabletail{}}%
8077 \begin{supertabular}{lp{\glsgdescwidth}lp{\glspagelistwidth}}}%
8078 {\end{supertabular}}}%
8079 }

```

**altsuper4colborder** The altsuper4colborder style is like the altsuper4col but with a border.

```

8080 \newglossarystyle{altsuper4colborder}{%
      Base it on the glostylesuper4colborder style:
8081 \setglossarystyle{super4colborder}%
      Put the glossary in a supertabular environment with four columns and a hori-
      zontal line in the head and tail:
8082 \renewenvironment{theglossary}%
8083 {\tablehead{\hline}\tabletail{\hline}%
8084 \begin{supertabular}%
8085 {\lllp{\glsgdescwidth}lllp{\glspagelistwidth}ll}%
8086 {\end{supertabular}}}%
8087 }

```

**per4colheaderborder** The altsuper4colheaderborder style is like the altsuper4col but with a header and border.

```

8088 \newglossarystyle{altsuper4colheaderborder}{%
      Base it on the glostylesuper4colheaderborder style:
8089 \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:
8090 \renewenvironment{theglossary}%
8091 {\tablehead{\hline
8092 \bfseries\entryname &
8093 \bfseries\descriptionname &
8094 \bfseries\symbolname &

```

```

8095      \bfseries\pagelistname\tabularnewline\hline}%
8096      \tabletail{\hline}%
8097      \begin{supertabular}%
8098          {\lllp{\glsdescwidth}\lllp{\glspagelistwidth}}}%
8099      {\end{supertabular}}%
8100 }

```

## 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.

```

8101 \ProvidesPackage{glossary-superragged}[2013/11/14 v4.0 (NLCT)]

```

Requires the package:

```

8102 \RequirePackage{array}

```

Requires the package:

```

8103 \RequirePackage{supertabular}

```

`\glsdescwidth` This is a length that governs the width of the description column. This may already have been defined.

```

8104 \@ifundefined{glsdescwidth}{%
8105     \newlength\glsdescwidth
8106     \setlength{\glsdescwidth}{0.6\hsize}
8107 }{}

```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

```

8108 \@ifundefined{glspagelistwidth}{%
8109     \newlength\glspagelistwidth
8110     \setlength{\glspagelistwidth}{0.1\hsize}
8111 }{}

```

`superragged` The superragged glossary style uses the supertabular environment.

```

8112 \newglossarystyle{superragged}{%

```

Put the glossary in a supertabular environment with two columns and no head or tail:

```

8113     \renewenvironment{theglossary}%
8114         {\tablehead{}\tabletail}%
8115         \begin{supertabular}{l>{\raggedright}p{\glsdescwidth}}}%
8116         {\end{supertabular}}%

```

Do nothing at the start of the table:

```

8117     \renewcommand*{\glossaryheader}{}%

```

No group headings:

```

8118     \renewcommand*{\glsgroupheading}[1]{}%

```

Main (level 0) entries put in a row (name in first column, description and page list in second column):

```
8119 \renewcommand{\glossentry}[2]{%
8120   \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
8121   \glossentrydesc{##1}\glspostdescription\space ##2%
8122   \tabularnewline
8123 }
```

Sub entries put in a row (no name, description and page list in second column):

```
8124 \renewcommand{\subglossentry}[3]{%
8125   &
8126   \glssubentryitem{##2}%
8127   \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space
8128   ##3%
8129   \tabularnewline
8130 }
```

Blank row between groups:

```
8131 \renewcommand*{\glsgroupskip}{\ifglsgnোগroupskip\else & \tabularnewline\fi}%
8132 }
```

superraggedborder The superraggedborder style is like the above, but with horizontal and vertical lines:

```
8133 \newglossarystyle{superraggedborder}{%
```

Base it on the glostylessuperragged style:

```
8134 \setglossarystyle{superragged}%
```

Put the glossary in a supertabular environment with two columns and a horizontal line in the head and tail:

```
8135 \renewenvironment{theglossary}%
8136   {\tablehead{\hline}\tabletail{\hline}%
8137   \begin{supertabular}{|l|>\raggedright}p{\glsgdescwidth}}}%
8138   {\end{supertabular}}%
8139 }
```

superraggedheader The superraggedheader style is like the super style, but with a header:

```
8140 \newglossarystyle{superraggedheader}{%
```

Base it on the glostylessuperragged style:

```
8141 \setglossarystyle{superragged}%
```

Put the glossary in a supertabular environment with two columns, a header and no tail:

```
8142 \renewenvironment{theglossary}%
8143   {\tablehead{\bfseries \entryname & \bfseries \descriptionname
8144     \tabularnewline}%
8145   \tabletail{}}%
8146   \begin{supertabular}{|l|>\raggedright}p{\glsgdescwidth}}}%
8147   {\end{supertabular}}%
8148 }
```

`rraggedheaderborder` The `superraggedheaderborder` style is like the `superragged` style but with a header and border:

```
8149 \newglossarystyle{superraggedheaderborder}{%
```

Base it on the `glostylesuper` style:

```
8150 \setglossarystyle{superragged}%
```

Put the glossary in a `supertabular` environment with two columns, a header and horizontal lines above and below the table:

```
8151 \renewenvironment{theglossary}%
8152 {\tablehead{\hline\bfseries \entryname &
8153 \bfseries \descriptionname\tabularnewline\hline}%
8154 \tabletail{\hline}
8155 \begin{supertabular}{|l|>\raggedright}p{\glsgdescwidth}|}%
8156 {\end{supertabular}}%
8157 }
```

`superragged3col` The `superragged3col` style is like the `superragged` style, but with 3 columns:

```
8158 \newglossarystyle{superragged3col}{%
```

Put the glossary in a `supertabular` environment with three columns and no head or tail:

```
8159 \renewenvironment{theglossary}%
8160 {\tablehead{}\tabletail{}}%
8161 \begin{supertabular}{|l>\raggedright}p{\glsgdescwidth}%
8162 >\raggedright}p{\glspagelistwidth}}}%
8163 {\end{supertabular}}%
```

Do nothing at the start of the table:

```
8164 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
8165 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
8166 \renewcommand{\glossentry}[2]{%
8167 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
8168 \glossentrydesc{##1} &
8169 ##2\tabularnewline
8170 }%
```

Sub entries on a row (no name, description in second column, page list in last column):

```
8171 \renewcommand{\subglossentry}[3]{%
8172 &
8173 \glssubentryitem{##2}%
8174 \glstarget{##2}{\strut}\glossentrydesc{##2} &
8175 ##3\tabularnewline
8176 }%
```

Blank row between groups:

```
8177 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & &\tabularnewline\fi}%  
8178 }
```

**superragged3colborder** The `superragged3colborder` style is like the `superragged3col` style, but with a border:

```
8179 \newglossarystyle{superragged3colborder}{%
```

Base it on the `glostylesuperragged3col` style:

```
8180 \setglossarystyle{superragged3col}%
```

Put the glossary in a `supertabular` environment with three columns and a horizontal line in the head and tail:

```
8181 \renewenvironment{theglossary}%  
8182 {\tablehead{\hline}\tabletail{\hline}%  
8183 \begin{supertabular}{|l|>\raggedright}p{\glsdescwidth}|%  
8184 >\raggedright}p{\glspagelistwidth}|}%  
8185 {\end{supertabular}}%  
8186 }
```

**superragged3colheader** The `superragged3colheader` style is like the `superragged3col` style but with a header row:

```
8187 \newglossarystyle{superragged3colheader}{%
```

Base it on the `glostylesuperragged3col` style:

```
8188 \setglossarystyle{superragged3col}%
```

Put the glossary in a `supertabular` environment with three columns, a header and no tail:

```
8189 \renewenvironment{theglossary}%  
8190 {\tablehead{\bfseries\entryname&\bfseries\descriptionname&  
8191 \bfseries\pagelistname\tabularnewline}\tabletail{}}%  
8192 \begin{supertabular}{|l>\raggedright}p{\glsdescwidth}%  
8193 >\raggedright}p{\glspagelistwidth}}}%  
8194 {\end{supertabular}}%  
8195 }
```

**superragged3colheaderborder** The `superragged3colheaderborder` style is like the `superragged3col` style but with a header and border:

```
8196 \newglossarystyle{superragged3colheaderborder}{%
```

Base it on the `glostylesuperragged3colborder` style:

```
8197 \setglossarystyle{superragged3colborder}%
```

Put the glossary in a `supertabular` environment with three columns, a header with horizontal lines and a horizontal line in the tail:

```
8198 \renewenvironment{theglossary}%  
8199 {\tablehead{\hline  
8200 \bfseries\entryname&\bfseries\descriptionname&  
8201 \bfseries\pagelistname\tabularnewline\hline}%
```

```

8202     \tabletail{\hline}%
8203     \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}|%
8204       >{\raggedright}p{\glspagelistwidth}|}%
8205     {\end{supertabular}}%
8206 }

```

`altsuperragged4col` The `altsuperragged4col` glossary style is like `altsuper4col` style in the package but uses ragged right formatting in the description and page list columns.

```

8207 \newglossarystyle{altsuperragged4col}{%

```

Put the glossary in a `supertabular` environment with four columns and no head or tail:

```

8208   \renewenvironment{theglossary}%
8209     {\tablehead{}\tabletail}%
8210     \begin{supertabular}{|l>{\raggedright}p{\glsgdescwidth}l%
8211       >{\raggedright}p{\glspagelistwidth}}}%
8212     {\end{supertabular}}%

```

Do nothing at the start of the table:

```

8213   \renewcommand*{\glossaryheader}{}%

```

No group headings:

```

8214   \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:

```

8215   \renewcommand{\glossentry}[2]{%
8216     \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
8217     \glossentrydesc{##1} &
8218     \glossentrysymbol{##1} & ##2\tabularnewline
8219   }%

```

Sub entries on a row with no name, the description in the second column, symbol in third column and page list in last column:

```

8220   \renewcommand{\subglossentry}[3]{%
8221     &
8222     \glssubentryitem{##2}%
8223     \glstarget{##2}{\strut}\glossentrydesc{##2} &
8224     \glossentrysymbol{##2} & ##3\tabularnewline
8225   }%

```

Blank row between groups:

```

8226   \renewcommand*{\glsgroupskip}{\ifglsgnোগroupskip\else & & \tabularnewline\fi}%
8227 }

```

`perragged4colheader` The `altsuperragged4colheader` style is like the `altsuperragged4col` style but with a header row.

```

8228 \newglossarystyle{altsuperragged4colheader}{%

```

Base it on the `glostylealtsuperragged4col` style:

```

8229   \setglossarystyle{altsuperragged4col}%

```

Put the glossary in a supertabular environment with four columns, a header and no tail:

```

8230 \renewenvironment{theglossary}%
8231   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
8232     \bfseries\symbolname &
8233     \bfseries\pagelistname\tabularnewline}\tabletail{}}%
8234   \begin{supertabular}{1>{\raggedright}p{\glsgdescwidth}1%
8235     >{\raggedright}p{\glspagelistwidth}}}%
8236   {\end{supertabular}}%
8237 }

```

`altperragged4colborder` The `altperragged4colborder` style is like the `altperragged4col` style but with a border.

```

8238 \newglossarystyle{altperragged4colborder}{%

```

Base it on the `glostylealtperragged4col` style:

```

8239 \setglossarystyle{altperragged4col}%

```

Put the glossary in a supertabular environment with four columns and a horizontal line in the head and tail:

```

8240 \renewenvironment{theglossary}%
8241   {\tablehead{\hline}\tabletail{\hline}%
8242   \begin{supertabular}%
8243     {\l|>{\raggedright}p{\glsgdescwidth}l|}%
8244     >{\raggedright}p{\glspagelistwidth}l}%
8245   {\end{supertabular}}%
8246 }

```

`altperragged4colheaderborder` The `altperragged4colheaderborder` style is like the `altperragged4col` style but with a header and border.

```

8247 \newglossarystyle{altperragged4colheaderborder}{%

```

Base it on the `glostylealtperragged4col` style:

```

8248 \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:

```

8249 \renewenvironment{theglossary}%
8250   {\tablehead{\hline
8251     \bfseries\entryname &
8252     \bfseries\descriptionname &
8253     \bfseries\symbolname &
8254     \bfseries\pagelistname\tabularnewline\hline}%
8255   \tabletail{\hline}%
8256   \begin{supertabular}%
8257     {\l|>{\raggedright}p{\glsgdescwidth}l|}%
8258     >{\raggedright}p{\glspagelistwidth}l}%
8259   {\end{supertabular}}%
8260 }

```

## 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.

```
8261 \ProvidesPackage{glossary-tree}[2014/07/30 v4.08 (NLCT)]
```

`\glstreenamefmt` Format used to display the name in the tree styles. (This may be counteracted by `\glsnamefont`.) This command is also used to format the group headings.

```
8262 \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.

```
8263 \newglossarystyle{index}{%
```

Set the paragraph indentation and skip and define `\item` to be the same as that used by the index:

```
8264 \renewenvironment{theglossary}{%
8265   {\setlength{\parindent}{0pt}%
8266    \setlength{\parskip}{0pt plus 0.3pt}%
8267    \let\item\@idxitem}%
8268   {\par}%
8269   \renewcommand*{\glossaryheader}{}}%
8270   \renewcommand*{\glsgroupheading}[1]{}%
8271   \renewcommand*{\glossentry}[2]{%
8272     \item\glstentryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
8273     \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
8274     \space \glossentrydesc{##1}\glspostdescription\space ##2%
8275   }%
```

Do nothing at the start of the environment:

```
8269 \renewcommand*{\glossaryheader}{}%
```

No group headers:

```
8270 \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.

```
8271 \renewcommand*{\glossentry}[2]{%
8272   \item\glstentryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
8273   \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
8274   \space \glossentrydesc{##1}\glspostdescription\space ##2%
8275 }
```

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.

```
8276 \renewcommand{\subglossentry}[3]{%
8277   \ifcase##1\relax
8278   % level 0
8279   \item
8280   \or
8281   % level 1
8282   \subitem
```



```

8283     \glssubentryitem{##2}%
8284     \else
8285     % all other levels
8286     \subsubitem
8287     \fi
8288     \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}%
8289     \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
8290     \space\glossentrydesc{##2}\glspostdescription\space ##3%
8291 }%

```

Vertical gap between groups is the same as that used by indices:

```

8292 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}}

```

**indexgroup** The indexgroup style is like the index style but has headings.

```

8293 \newglossarystyle{indexgroup}{%

```

Base it on the glostyleindex style:

```

8294 \setglossarystyle{index}%

```

Add a heading for each group. This puts the group's title in bold followed by a vertical gap.

```

8295 \renewcommand*{\glsgroupheading}[1]{%
8296 \item\glstreenamefmt{\glsgetgrouptitle{##1}}\indexspace}%
8297 }

```

**indexhypergroup** The indexhypergroup style is like the indexgroup style but has hyper navigation.

```

8298 \newglossarystyle{indexhypergroup}{%

```

Base it on the glostyleindex style:

```

8299 \setglossarystyle{index}%

```

Put navigation links to the groups at the start of the glossary:

```

8300 \renewcommand*{\glossaryheader}{%
8301 \item\glstreenamefmt{\glslnavigation}\indexspace}%

```

Add a heading for each group (with a target). The group's title is in bold followed by a vertical gap.

```

8302 \renewcommand*{\glsgroupheading}[1]{%
8303 \item\glstreenamefmt{\glslnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
8304 \indexspace}%
8305 }

```

**tree** The tree glossary style is similar in style to the index style, but can have arbitrary levels.

```

8306 \newglossarystyle{tree}{%

```

Set the paragraph indentation and skip:

```

8307 \renewenvironment{theglossary}%
8308 {\setlength{\parindent}{0pt}%
8309 \setlength{\parskip}{0pt plus 0.3pt}}%
8310 {}%

```

Do nothing at the start of the theglossary environment:

```
8311 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
8312 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries: name in bold, followed by symbol in brackets (if it exists), the description and the page list:

```
8313 \renewcommand{\glossentry}[2]{%
8314   \hangindent0pt\relax
8315   \parindent0pt\relax
8316   \glstryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
8317   \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
8318   \space\glossentrydesc{##1}\glspostdescription\space##2\par
8319 }%
```

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.

```
8320 \renewcommand{\subglossentry}[3]{%
8321   \hangindent##1\glstreeindent\relax
8322   \parindent##1\glstreeindent\relax
8323   \ifnum##1=1\relax
8324     \glssubentryitem{##2}%
8325   \fi
8326   \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}%
8327   \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
8328   \space\glossentrydesc{##2}\glspostdescription\space ##3\par
8329 }%
```

Vertical gap between groups is the same as that used by indices:

```
8330 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}}
```

**treegroup** Like the tree style but the glossary groups have headings.

```
8331 \newglossarystyle{treegroup}{%
```

Base it on the glostyletree style:

```
8332 \setglossarystyle{tree}%
```

Each group has a heading (in bold) followed by a vertical gap):

```
8333 \renewcommand{\glsgroupheading}[1]{\par
8334   \noindent\glstreenamefmt{\glsgrouptitle{##1}}\par\indexspace}%
8335 }
```

**treehypergroup** The treehypergroup style is like the treegroup style, but has a set of links to the groups at the start of the glossary.

```
8336 \newglossarystyle{treehypergroup}{%
```

Base it on the glostyletree style:

```
8337 \setglossarystyle{tree}%
```

Put navigation links to the groups at the start of the theglossary environment:

```
8338 \renewcommand*{\glossaryheader}{%
8339 \par\noindent\glstreenamefmt{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
8340 \renewcommand*{\glsgroupheading}[1]{%
8341 \par\noindent
8342 \glstreenamefmt{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}\par
8343 \indexspace}%
8344 }
```

`\glstreeindent` Length governing left indent for each level of the tree style.

```
8345 \newlength\glstreeindent
8346 \setlength{\glstreeindent}{10pt}
```

`treenoname` The `treenoname` glossary style is like the tree style, but doesn't print the name or symbol for sub-levels.

```
8347 \newglossarystyle{treenoname}{%
```

Set the paragraph indentation and skip:

```
8348 \renewenvironment{theglossary}%
8349 {\setlength{\parindent}{0pt}%
8350 \setlength{\parskip}{0pt plus 0.3pt}}%
8351 {}%
```

No header:

```
8352 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
8353 \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.

```
8354 \renewcommand{\glossentry}[2]{%
8355 \hangindent0pt\relax
8356 \parindent0pt\relax
8357 \glstryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
8358 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
8359 \space\glossentrydesc{##1}\glspostdescription\space##2\par
8360 }%
```

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.

```
8361 \renewcommand{\subglossentry}[3]{%
8362 \hangindent##1\glstreeindent\relax
8363 \parindent##1\glstreeindent\relax
8364 \ifnum##1=1\relax
8365 \glssubentryitem{##2}%
8366 \fi
8367 \glstarget{##2}{\strut}%
8368 \glossentrydesc{##2}\glspostdescription\space##3\par
8369 }%
```

Vertical gap between groups is the same as that used by indices:

```
8370 \renewcommand*{\glsgroupskip}{\ifglsgroupskip\else\indexspace\fi}%
8371 }
```

**treenonamegroup** Like the `treenoname` style but the glossary groups have headings.

```
8372 \newglossarystyle{treenonamegroup}{%
```

Base it on the `glostyletreenoname` style:

```
8373 \setglossarystyle{treenoname}%
```

Give each group a heading:

```
8374 \renewcommand{\glsgroupheading}[1]{\par
```

```
8375 \noindent\glstreenamefmt{\glsgrouptitle{##1}}\par\indexspace}%
```

```
8376 }
```

**treenonamehypergroup** The `treenonamehypergroup` style is like the `treenonamegroup` style, but has a set of links to the groups at the start of the glossary.

```
8377 \newglossarystyle{treenonamehypergroup}{%
```

Base it on the `glostyletreenoname` style:

```
8378 \setglossarystyle{treenoname}%
```

Put navigation links to the groups at the start of the `theglossary` environment:

```
8379 \renewcommand*{\glossaryheader}{%
```

```
8380 \par\noindent\glstreenamefmt{\glsgroupnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
8381 \renewcommand*{\glsgroupheading}[1]{%
```

```
8382 \par\noindent
```

```
8383 \glstreenamefmt{\glsgrouphypertarget{##1}{\glsgrouptitle{##1}}}\par
```

```
8384 \indexspace}%
```

```
8385 }
```

**\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.

```
8386 \newcommand*{\glissetwidest}[2][0]{%
```

```
8387 \expandafter\def\csname @glswidestname\romannumeral#1\endcsname{%
```

```
8388 #2}%
```

```
8389 }
```

**\@glswidestname** Initialise `\@glswidestname`.

```
8390 \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`.

```
8391 \newglossarystyle{alttree}{%
```

Redefine theglossary environment.

```
8392 \renewenvironment{theglossary}%
8393   {\def\@gls@prevlevel{-1}%
8394    \mbox{}\par}%
8395   {\par}%
```

Set the header and group headers to nothing.

```
8396 \renewcommand*\glossaryheader{}%
8397 \renewcommand*\glsgroupheading[1]{}%
```

Redefine the way that the level 0 entries are displayed.

```
8398 \renewcommand{\glosentry}[2]{%
8399   \ifnum\@gls@prevlevel=0\relax
8400   \else
```

Find out how big the indentation should be by measuring the widest entry.

```
8401     \settowidth{\glstreeindent}{\glstreenamfmt{\@glswidestname\space}}%
8402   \fi
```

Set the hangindent and paragraph indent.

```
8403     \hangindent\glstreeindent
8404     \parindent\glstreeindent
```

Put the name to the left of the paragraph block.

```
8405     \makebox[0pt][r]{\makebox[\glstreeindent][l]{%
8406       \glsentryitem{##1}\glstreenamfmt{\glstarget{##1}{\glosentryname{##1}}}}}%
```

If the symbol is missing, ignore it, otherwise put it in brackets.

```
8407     \ifglshassymbol{##1}{\space(\glosentrysymbol{##1})}{}%
```

Do the description followed by the description terminator and location list.

```
8408     \glosentrydesc{##1}\glspostdescription \space ##2\par
```

Set the previous level to 0.

```
8409     \def\@gls@prevlevel{0}%
8410   }%
```

Redefine the way sub-entries are displayed.

```
8411 \renewcommand{\subglosentry}[3]{%
```

Increment and display the sub-entry counter if this is a level 1 entry and the sub-entry counter is in use.

```
8412   \ifnum##1=1\relax
8413     \glssubentryitem{##2}%
8414   \fi
```

If the level hasn't changed, keep the same settings, otherwise adjust \glstreeindent accordingly.

```
8415   \ifnum\@gls@prevlevel=##1\relax
8416   \else
```

Compute the widest entry for this level, or for level 0 if not defined for this level.  
Store in \gls@tmplen

```

8417     \@ifundefined{glswidestname\romannumeral##1}{%
8418         \settowidth{gls@tmplen}{\glstreenamefmt{\glswidestname\space}}}%
8419         \settowidth{gls@tmplen}{\glstreenamefmt{%
8420             \csname glswidestname\romannumeral##1\endcsname\space}}}%

```

Determine if going up or down a level

```

8421     \ifnum\gls@prevlevel<##1\relax

```

Depth has increased, so add the width of the widest entry to \glstreeindent.

```

8422         \setlength\glstreeindent\gls@tmplen
8423         \addtolength\glstreeindent\parindent
8424         \parindent\glstreeindent
8425     \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.

```

8426         \@ifundefined{glswidestname\romannumeral\gls@prevlevel}{%
8427             \settowidth{\glstreeindent}{\glstreenamefmt{%
8428                 \glswidestname\space}}}%
8429             \settowidth{\glstreeindent}{\glstreenamefmt{%
8430                 \csname glswidestname\romannumeral\gls@prevlevel
8431                     \endcsname\space}}}%

```

Subtract this length from the previous level's paragraph indent and set to \glstreeindent.

```

8432         \addtolength\parindent{-\glstreeindent}%
8433         \setlength\glstreeindent\parindent
8434     \fi
8435 \fi

```

Set the hanging indentation.

```

8436     \hangindent\glstreeindent

```

Put the name to the left of the paragraph block

```

8437     \makebox[0pt][r]{\makebox[\gls@tmplen][l]{%
8438         \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}}}%

```

If the symbol is missing, ignore it, otherwise put it in brackets.

```

8439     \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%

```

Do the description followed by the description terminator and location list.

```

8440     \glossentrydesc{##2}\glspostdescription\space ##3\par

```

Set the previous level macro to the current level.

```

8441     \def\gls@prevlevel{##1}%
8442 }%

```

Vertical gap between groups is the same as that used by indices:

```

8443 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
8444 }

```

**alttreegroup** Like the `alttree` style but the glossary groups have headings.

```
8445 \newglossarystyle{alttreegroup}{%
      Base it on the glostylealttree style:
8446   \setglossarystyle{alttree}%
      Give each group a heading.
8447   \renewcommand{\glsgroupheading}[1]{\par
8448     \def\@gls@prevlevel{-1}%
8449     \hangindent0pt\relax
8450     \parindent0pt\relax
8451     \glstreenamefmt{\glsgetgrouptitle{##1}}\par\indexspace}%
8452 }
```

**alttreehypergroup** The `alttreehypergroup` style is like the `alttreegroup` style, but has a set of links to the groups at the start of the glossary.

```
8453 \newglossarystyle{alttreehypergroup}{%
      Base it on the glostylealttree style:
8454   \setglossarystyle{alttree}%
      Put the navigation links in the header
8455   \renewcommand*{\glossaryheader}{%
8456     \par
8457     \def\@gls@prevlevel{-1}%
8458     \hangindent0pt\relax
8459     \parindent0pt\relax
8460     \glstreenamefmt{\glsnavigation}\par\indexspace}%
      Put a hypertarget at the start of each group
8461   \renewcommand*{\glsgroupheading}[1]{%
8462     \par
8463     \def\@gls@prevlevel{-1}%
8464     \hangindent0pt\relax
8465     \parindent0pt\relax
8466     \glstreenamefmt{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
8467     \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.

```
8468 \NeedsTeXFormat{LaTeX2e}
8469 \ProvidesPackage{glossaries-compatible-207}[2011/04/02 v1.0 (NLCT)]
```

**\GlsAddXdyAttribute** Adds an attribute in old format.

```
8470 \ifglsxindy
8471   \renewcommand*\GlsAddXdyAttribute[1]{%
```

```

8472 \edef\@xdyattributes{\@xdyattributes ^^J \string"#1\string"}%
8473 \expandafter\toks@\expandafter{\@xdylocref}%
8474 \edef\@xdylocref{\the\toks@ ^^J%
8475 (markup-locref
8476 :open \string"\string~n\string\setentrycounter
8477 {\noexpand\glscounter}%
8478 \expandafter\string\csname#1\endcsname
8479 \expandafter\@gobble\string\{\string" ^^J
8480 :close \string"\expandafter\@gobble\string\}\string" ^^J
8481 :attr \string"#1\string")}}

```

Only has an effect before \writeist:

```
8482 \fi
```

\GlsAddXdyCounters

```

8483 \renewcommand*\GlsAddXdyCounters[1]{%
8484 \GlossariesWarning{\string\GlsAddXdyCounters\space not available
8485 in compatibility mode.}%
8486 }

```

Add predefined attributes

```

8487 \GlsAddXdyAttribute{glsnumberformat}
8488 \GlsAddXdyAttribute{textrm}
8489 \GlsAddXdyAttribute{textsf}
8490 \GlsAddXdyAttribute{texttt}
8491 \GlsAddXdyAttribute{textbf}
8492 \GlsAddXdyAttribute{textmd}
8493 \GlsAddXdyAttribute{textit}
8494 \GlsAddXdyAttribute{textup}
8495 \GlsAddXdyAttribute{textsl}
8496 \GlsAddXdyAttribute{textsc}
8497 \GlsAddXdyAttribute{emph}
8498 \GlsAddXdyAttribute{glshypernumber}
8499 \GlsAddXdyAttribute{hyperrrm}
8500 \GlsAddXdyAttribute{hypersf}
8501 \GlsAddXdyAttribute{hypertt}
8502 \GlsAddXdyAttribute{hyperbf}
8503 \GlsAddXdyAttribute{hypermd}
8504 \GlsAddXdyAttribute{hyperit}
8505 \GlsAddXdyAttribute{hyperup}
8506 \GlsAddXdyAttribute{hypersl}
8507 \GlsAddXdyAttribute{hypersc}
8508 \GlsAddXdyAttribute{hyperemph}

```

\GlsAddXdyLocation Restore v2.07 definition:

```

8509 \ifglxindy
8510 \renewcommand*\GlsAddXdyLocation[2]{%
8511 \edef\@xdyuserlocationdefs{%
8512 \@xdyuserlocationdefs ^^J%

```



```

8513      (define-location-class \string"#1\string"^^J\space\space
8514      \space(#2))
8515    }%
8516    \edef\xdyuserlocationnames{%
8517      \xdyuserlocationnames^^J\space\space\space
8518      \string"#1\string"}%
8519  }
8520 \fi

```

\@do@wrglossary

```

8521 \renewcommand{\@do@wrglossary}[1]{%

```

Determine whether to use xindy or makeindex syntax

```

8522 \ifglxindy

```

Need to determine if the formatting information starts with a ( or ) indicating a range.

```

8523 \expandafter\@glo@check@mkidxrangechar\@glsnumberformat\@nil
8524 \def\@glo@range{%
8525 \expandafter\if\@glo@prefix(\relax
8526 \def\@glo@range{:open-range}%
8527 \else
8528 \expandafter\if\@glo@prefix)\relax
8529 \def\@glo@range{:close-range}%
8530 \fi
8531 \fi

```

Get the location and escape any special characters

```

8532 \protected@edef\@glslocref{\theglentrycounter}%
8533 \@gls@checkmkidxchars\@glslocref

```

Write to the glossary file using xindy syntax.

```

8534 \glossary[\csname glo@#1@type\endcsname]{%
8535 (indexentry :tkey (\csname glo@#1@index\endcsname)
8536 :locref \string"\@glslocref\string" %
8537 :attr \string"\@glo@suffix\string" \@glo@range
8538 )
8539 }%
8540 \else

```

Convert the format information into the format required for makeindex

```

8541 \@set@glo@numformat\@glo@numfmt\@gls@counter\@glsnumberformat

```

Write to the glossary file using makeindex syntax.

```

8542 \glossary[\csname glo@#1@type\endcsname]{%
8543 \string\glossaryentry{\csname glo@#1@index\endcsname
8544 \@gls@encapchar\@glo@numfmt}\theglentrycounter}}%
8545 \fi
8546 }

```

\@set@glo@numformat Only had 3 arguments in v2.07

```

8547 \def\@set@glo@numformat#1#2#3{%
8548   \expandafter\@glo@check@mkidxrangechar#3\@nil
8549   \protected@edef#1{%
8550     \@glo@prefix setentrycounter[]{#2}%
8551     \expandafter\string\csname\@glo@suffix\endcsname
8552   }%
8553   \@gls@checkmkidxchars#1%
8554 }

```

\writeist    Redefine \writeist back to the way it was in v2.07, but change \istfile to  
\glswrite.

```

8555 \ifglxindy
8556   \def\writeist{%
8557     \openout\glswrite=\istfilename
8558     \write\glswrite{;; xindy style file created by the glossaries
8559       package in compatible-2.07 mode}%
8560     \write\glswrite{;; for document '\jobname' on
8561       \the\year-\the\month-\the\day}%
8562     \write\glswrite{^^J; required styles^^J}
8563     \for\@xdystyle:=\@xdyrequiredstyles\do{%
8564       \ifx\@xdystyle\@empty
8565         \else
8566           \protected@write\glswrite{{(require
8567             \string"\@xdystyle.xdy\string")}}%
8568       \fi
8569     }%
8570     \write\glswrite{^^J
8571       ; list of allowed attributes (number formats)^^J}%
8572     \write\glswrite{(define-attributes ((\@xdyattributes)))}%
8573     \write\glswrite{^^J; user defined alphabets^^J}%
8574     \write\glswrite{\@xdyuseralphabets}%
8575     \write\glswrite{^^J; location class definitions^^J}%
8576     \protected@edef\@gls@roman{\@roman{0}\string"
8577       \string"roman-numbers-lowercase\string" :sep \string"}}%
8578     \@onelevel@sanitize\@gls@roman
8579     \edef\@tmp{\string" \string"roman-numbers-lowercase\string"
8580       :sep \string"}%
8581     \@onelevel@sanitize\@tmp
8582     \ifx\@tmp\@gls@roman
8583       \write\glswrite{(define-location-class
8584         \string"roman-page-numbers\string"^^J\space\space\space
8585         (\string"roman-numbers-lowercase\string")
8586         :min-range-length \@glsminrange)}}%
8587     \else
8588       \write\glswrite{(define-location-class
8589         \string"roman-page-numbers\string"^^J\space\space\space
8590         (:sep "\@gls@roman")
8591         :min-range-length \@glsminrange)}}%
8592     \fi

```

```

8593 \write\glswrite{(define-location-class
8594   \string"Roman-page-numbers\string"^^J\space\space\space
8595   (\string"roman-numbers-uppercase\string")
8596   :min-range-length \@glsmminrange)}}%
8597 \write\glswrite{(define-location-class
8598   \string"arabic-page-numbers\string"^^J\space\space\space
8599   (\string"arabic-numbers\string")
8600   :min-range-length \@glsmminrange)}}%
8601 \write\glswrite{(define-location-class
8602   \string"alpha-page-numbers\string"^^J\space\space\space
8603   (\string"alpha\string")
8604   :min-range-length \@glsmminrange)}}%
8605 \write\glswrite{(define-location-class
8606   \string"Alpha-page-numbers\string"^^J\space\space\space
8607   (\string"ALPHA\string")
8608   :min-range-length \@glsmminrange)}}%
8609 \write\glswrite{(define-location-class
8610   \string"Appendix-page-numbers\string"^^J\space\space\space
8611   (\string"ALPHA\string"
8612   :sep \string"\@glAlphacompositor\string"
8613   \string"arabic-numbers\string")
8614   :min-range-length \@glsmminrange)}}%
8615 \write\glswrite{(define-location-class
8616   \string"arabic-section-numbers\string"^^J\space\space\space
8617   (\string"arabic-numbers\string"
8618   :sep \string"\glsc compositor\string"
8619   \string"arabic-numbers\string")
8620   :min-range-length \@glsmminrange)}}%
8621 \write\glswrite{^^J; user defined location classes}%
8622 \write\glswrite{@x dyuserlocationdefs}%
8623 \write\glswrite{^^J; define cross-reference class^^J}%
8624 \write\glswrite{(define-crossref-class \string"see\string"
8625   :unverified )}%
8626 \write\glswrite{(markup-crossref-list
8627   :class \string"see\string"^^J\space\space\space
8628   :open \string"\string\glseeformat\string"
8629   :close \string"{}\string")}%
8630 \write\glswrite{^^J; define the order of the location classes}%
8631 \write\glswrite{(define-location-class-order
8632   (\@x dylocationclassorder))}%
8633 \write\glswrite{^^J; define the glossary markup^^J}%
8634 \write\glswrite{(markup-index^^J\space\space\space
8635   :open \string"\string
8636   \glossarysection[\string\glossarytoctitle]{\string
8637   \glossarytitle}\string\glossarypreamble\string~n\string\begin
8638   {theglossary}\string\glossaryheader\string~n\string" ^^J\space
8639   \space\space:close \string"\expandafter\@gobble
8640   \string\%\string~n\string
8641   \end{theglossary}\string\glossarypostamble

```

```

8642     \string~n\string" ^^J\space\space\space
8643     :tree)}}%
8644     \write\glswrite{(markup-letter-group-list
8645     :sep \string"\string\glsgroupskip\string~n\string"))}%
8646     \write\glswrite{(markup-indexentry
8647     :open \string"\string\relax \string\glsresetentrylist
8648     \string~n\string"))}%
8649     \write\glswrite{(markup-locclass-list :open
8650     \string"\glsopenbrace\string\glossaryentrynumbers
8651     \glsopenbrace\string\relax\space \string"^^J\space\space\space
8652     :sep \string", \string"
8653     :close \string"\glsclosebrace\glsclosebrace\string"))}%
8654     \write\glswrite{(markup-locref-list
8655     :sep \string"\string\delimN\space\string"))}%
8656     \write\glswrite{(markup-range
8657     :sep \string"\string\delimR\space\string"))}%
8658     \@onelevel@sanitize\gls@suffixF
8659     \@onelevel@sanitize\gls@suffixFF
8660     \ifx\gls@suffixF\@empty
8661     \else
8662     \write\glswrite{(markup-range
8663     :close "\gls@suffixF" :length 1 :ignore-end)}}%
8664     \fi
8665     \ifx\gls@suffixFF\@empty
8666     \else
8667     \write\glswrite{(markup-range
8668     :close "\gls@suffixFF" :length 2 :ignore-end)}}%
8669     \fi
8670     \write\glswrite{^^J; define format to use for locations^^J}%
8671     \write\glswrite{@xdylocref}%
8672     \write\glswrite{^^J; define letter group list format^^J}%
8673     \write\glswrite{(markup-letter-group-list
8674     :sep \string"\string\glsgroupskip\string~n\string"))}%
8675     \write\glswrite{^^J; letter group headings^^J}%
8676     \write\glswrite{(markup-letter-group
8677     :open-head \string"\string\glsgroupheading
8678     \glsopenbrace\string"^^J\space\space\space
8679     :close-head \string"\glsclosebrace\string"))}%
8680     \write\glswrite{^^J; additional letter groups^^J}%
8681     \write\glswrite{@xdylettergroups}%
8682     \write\glswrite{^^J; additional sort rules^^J}%
8683     \write\glswrite{@xdysortrules}%
8684     \noist}
8685 \else
8686     \edef\@gls@actualchar{\string?}
8687     \edef\@gls@encapchar{\string|}
8688     \edef\@gls@levelchar{\string!}
8689     \edef\@gls@quotechar{\string"}
8690     \def\writeist{\relax

```

```

8691 \openout\glswrite=\istfilename
8692 \write\glswrite{\expandafter\@gobble\string\% makeindex style file
8693   created by the glossaries package}
8694 \write\glswrite{\expandafter\@gobble\string\% for document
8695   '\jobname' on \the\year-\the\month-\the\day}
8696 \write\glswrite{actual '\@gls@actualchar'}
8697 \write\glswrite{encap '\@gls@encapchar'}
8698 \write\glswrite{level '\@gls@levelchar'}
8699 \write\glswrite{quote '\@gls@quotechar'}
8700 \write\glswrite{keyword \string"\string\glossaryentry\string"}
8701 \write\glswrite{preamble \string"\string\glossarysection[\string
8702   \glossarytoctitle]{\string\glossarytitle}\string
8703   \glossarypreamble\string\n\string\begin{theglossary}\string
8704   \glossaryheader\string\n\string"}
8705 \write\glswrite{postamble \string"\string%\string\n\string
8706   \end{theglossary}\string\glossarypostamble\string\n
8707   \string"}
8708 \write\glswrite{group_skip \string"\string\glsgroupskip\string\n
8709   \string"}
8710 \write\glswrite{item_0 \string"\string%\string\n\string"}
8711 \write\glswrite{item_1 \string"\string%\string\n\string"}
8712 \write\glswrite{item_2 \string"\string%\string\n\string"}
8713 \write\glswrite{item_01 \string"\string%\string\n\string"}
8714 \write\glswrite{item_x1
8715   \string"\string\relax \string\glsresetentrylist\string\n
8716   \string"}
8717 \write\glswrite{item_12 \string"\string%\string\n\string"}
8718 \write\glswrite{item_x2
8719   \string"\string\relax \string\glsresetentrylist\string\n
8720   \string"}
8721 \write\glswrite{delim_0 \string"\string\{\string
8722   \glossaryentrynumbers\string\{\string\relax \string"}
8723 \write\glswrite{delim_1 \string"\string\{\string
8724   \glossaryentrynumbers\string\{\string\relax \string"}
8725 \write\glswrite{delim_2 \string"\string\{\string
8726   \glossaryentrynumbers\string\{\string\relax \string"}
8727 \write\glswrite{delim_t \string"\string\}\string\}\string"}
8728 \write\glswrite{delim_n \string"\string\delimN \string"}
8729 \write\glswrite{delim_r \string"\string\delimR \string"}
8730 \write\glswrite{headings_flag 1}
8731 \write\glswrite{heading_prefix
8732   \string"\string\glsgroupheading\string\{\string"}
8733 \write\glswrite{heading_suffix
8734   \string"\string\}\string\relax
8735   \string\glsresetentrylist \string"}
8736 \write\glswrite{symhead_positive \string"glssymbols\string"}
8737 \write\glswrite{numhead_positive \string"glslnumbers\string"}
8738 \write\glswrite{page_compositor \string"glscpositor\string"}
8739 \@gls@escbsdq\gls@suffixF

```

```

8740 \@gls@escbsdq\gls@suffixFF
8741 \ifx\gls@suffixF\@empty
8742 \else
8743 \write\glswrite{suffix_2p \string"\gls@suffixF\string"}
8744 \fi
8745 \ifx\gls@suffixFF\@empty
8746 \else
8747 \write\glswrite{suffix_3p \string"\gls@suffixFF\string"}
8748 \fi
8749 \noist
8750 }
8751 \fi

```

\noist

```
8752 \renewcommand*{\noist}{\let\writeist\relax}
```

Compatibility macros.

```

8753 \NeedsTeXFormat{LaTeX2e}
8754 \ProvidesPackage{glossaries-compatible-307}[2013/11/14 v4.0 (NLCT)]

```

Compatibility macros for predefined glossary styles:

**compatglossarystyle** Defines a compatibility glossary style.

```

8755 \newcommand{\compatglossarystyle}[2]{%
8756 \ifcsundef{@glscompstyle@#1}%
8757 {%
8758 \csdef{@glscompstyle@#1}{#2}%
8759 }%
8760 {%
8761 \PackageError{glossaries}{Glossary compatibility style ‘#1’ is already defined}{}%
8762 }%
8763 }

```

Backward compatible inline style.

```

8764 \compatglossarystyle{inline}{%
8765 \renewcommand{\glossaryentryfield}[5]{%
8766 \glsinlinedopostchild
8767 \gls@inlinesep
8768 \def\glo@desc{##3}%
8769 \def\@no@post@desc{\nopostdesc}%
8770 \glsentryitem{##1}\glsinlinenameformat{##1}{##2}%
8771 \ifx\glo@desc\@no@post@desc
8772 \glsinlineemptydescformat{##4}{##5}%
8773 \else
8774 \ifstrempy{##3}%
8775 {\glsinlineemptydescformat{##4}{##5}}%
8776 {\glsinlinedescformat{##3}{##4}{##5}}%
8777 \fi
8778 \ifglshaschildren{##1}%
8779 {%

```

```

8780      \glsresetsubentrycounter
8781      \glsinlineparentchildseparator
8782      \def\gls@inlinesubsep{}%
8783      \def\gls@inlinepostchild{\glsinlinepostchild}%
8784      }%
8785      {}%
8786      \def\gls@inlinesep{\glsinlineseparator}%
8787      }%

Sub-entries display description:
8788      \renewcommand{\glossarysubentryfield}[6]{%
8789          \gls@inlinesubsep%
8790          \glsinlinesubnameformat{##2}{##3}%
8791          \glssubentryitem{##2}\glsinlinesubdescformat{##4}{##5}{##6}%
8792          \def\gls@inlinesubsep{\glsinlinesubseparator}%
8793      }%
8794 }

Backward compatible list style.
8795 \compatglossarystyle{list}{%
8796     \renewcommand*{\glossaryentryfield}[5]{%
8797         \item[\glsentryitem{##1}\glstarget{##1}{##2}]
8798         ##3\glspostdescription\space ##5}%

Sub-entries continue on the same line:
8799     \renewcommand*{\glossarysubentryfield}[6]{%
8800         \glssubentryitem{##2}%
8801         \glstarget{##2}{\strut}##4\glspostdescription\space ##6.}%
8802 }

Backward compatible listgroup style.
8803 \compatglossarystyle{listgroup}{%
8804     \csuse{@glscompstyle@list}%
8805 }%

Backward compatible listhypergroup style.
8806 \compatglossarystyle{listhypergroup}{%
8807     \csuse{@glscompstyle@list}%
8808 }%

Backward compatible altlist style.
8809 \compatglossarystyle{altlist}{%
8810     \renewcommand*{\glossaryentryfield}[5]{%
8811         \item[\glsentryitem{##1}\glstarget{##1}{##2}]%
8812         \mbox{}\par\nobreak\@afterheading
8813         ##3\glspostdescription\space ##5}%
8814     \renewcommand{\glossarysubentryfield}[6]{%
8815         \par
8816         \glssubentryitem{##2}%
8817         \glstarget{##2}{\strut}##4\glspostdescription\space ##6}%
8818 }%

```

Backward compatible altlistgroup style.

```
8819 \compatglossarystyle{altlistgroup}{%  
8820 \csuse{@glscompstyle@altlist}%  
8821 }%
```

Backward compatible altlisthypergroup style.

```
8822 \compatglossarystyle{altlisthypergroup}{%  
8823 \csuse{@glscompstyle@altlist}%  
8824 }%
```

Backward compatible listdotted style.

```
8825 \compatglossarystyle{listdotted}{%  
8826 \renewcommand*{\glossaryentryfield}[5]{%  
8827 \item[]\makebox[\glslistdottedwidth][l]{%  
8828 \glentryitem{##1}\glstarget{##1}{##2}%  
8829 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##3}%  
8830 \renewcommand*{\glossarysubentryfield}[6]{%  
8831 \item[]\makebox[\glslistdottedwidth][l]{%  
8832 \glssubentryitem{##2}%  
8833 \glstarget{##2}{##3}%  
8834 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##4}%  
8835 }%
```

Backward compatible sublistdotted style.

```
8836 \compatglossarystyle{sublistdotted}{%  
8837 \csuse{@glscompstyle@listdotted}%  
8838 \renewcommand*{\glossaryentryfield}[5]{%  
8839 \item[\glentryitem{##1}\glstarget{##1}{##2}]}%  
8840 }%
```

Backward compatible long style.

```
8841 \compatglossarystyle{long}{%  
8842 \renewcommand*{\glossaryentryfield}[5]{%  
8843 \glentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\}%  
8844 \renewcommand*{\glossarysubentryfield}[6]{%  
8845 &  
8846 \glssubentryitem{##2}%  
8847 \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\}%  
8848 }%
```

Backward compatible longborder style.

```
8849 \compatglossarystyle{longborder}{%  
8850 \csuse{@glscompstyle@long}%  
8851 }%
```

Backward compatible longheader style.

```
8852 \compatglossarystyle{longheader}{%  
8853 \csuse{@glscompstyle@long}%  
8854 }%
```

Backward compatible longheaderborder style.

```
8855 \compatglossarystyle{longheaderborder}{%
```



```
8856 \csuse{@glscompstyle@long}%
8857 }%
```

Backward compatible long3col style.

```
8858 \compatglossarystyle{long3col}{%
8859   \renewcommand*{\glossaryentryfield}[5]{%
8860     \glsentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\\}%
8861   \renewcommand*{\glossarysubentryfield}[6]{%
8862     &
8863     \glssubentryitem{##2}%
8864     \glstarget{##2}{\strut}##4 & ##6\\}%
8865 }%
```

Backward compatible long3colborder style.

```
8866 \compatglossarystyle{long3colborder}{%
8867   \csuse{@glscompstyle@long3col}%
8868 }%
```

Backward compatible long3colheader style.

```
8869 \compatglossarystyle{long3colheader}{%
8870   \csuse{@glscompstyle@long3col}%
8871 }%
```

Backward compatible long3colheaderborder style.

```
8872 \compatglossarystyle{long3colheaderborder}{%
8873   \csuse{@glscompstyle@long3col}%
8874 }%
```

Backward compatible long4col style.

```
8875 \compatglossarystyle{long4col}{%
8876   \renewcommand*{\glossaryentryfield}[5]{%
8877     \glsentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\\}%
8878   \renewcommand*{\glossarysubentryfield}[6]{%
8879     &
8880     \glssubentryitem{##2}%
8881     \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%
8882 }%
```

Backward compatible long4colheader style.

```
8883 \compatglossarystyle{long4colheader}{%
8884   \csuse{@glscompstyle@long4col}%
8885 }%
```

Backward compatible long4colborder style.

```
8886 \compatglossarystyle{long4colborder}{%
8887   \csuse{@glscompstyle@long4col}%
8888 }%
```

Backward compatible long4colheaderborder style.

```
8889 \compatglossarystyle{long4colheaderborder}{%
8890   \csuse{@glscompstyle@long4col}%
8891 }%
```

Backward compatible altlong4col style.

```
8892 \compatglossarystyle{altlong4col}{%  
8893 \csuse{@glscompstyle@long4col}%  
8894 }%
```

Backward compatible altlong4colheader style.

```
8895 \compatglossarystyle{altlong4colheader}{%  
8896 \csuse{@glscompstyle@long4col}%  
8897 }%
```

Backward compatible altlong4colborder style.

```
8898 \compatglossarystyle{altlong4colborder}{%  
8899 \csuse{@glscompstyle@long4col}%  
8900 }%
```

Backward compatible altlong4colheaderborder style.

```
8901 \compatglossarystyle{altlong4colheaderborder}{%  
8902 \csuse{@glscompstyle@long4col}%  
8903 }%
```

Backward compatible long style.

```
8904 \compatglossarystyle{longragged}{%  
8905 \renewcommand*{\glossaryentryfield}[5]{%  
8906 \glstryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%  
8907 \tabularnewline}%  
8908 \renewcommand*{\glossarysubentryfield}[6]{%  
8909 &  
8910 \glssubentryitem{##2}%  
8911 \glstarget{##2}{\strut}##4\glspostdescription\space ##6%  
8912 \tabularnewline}%  
8913 }%
```

Backward compatible longraggedborder style.

```
8914 \compatglossarystyle{longraggedborder}{%  
8915 \csuse{@glscompstyle@longragged}%  
8916 }%
```

Backward compatible longraggedheader style.

```
8917 \compatglossarystyle{longraggedheader}{%  
8918 \csuse{@glscompstyle@longragged}%  
8919 }%
```

Backward compatible longraggedheaderborder style.

```
8920 \compatglossarystyle{longraggedheaderborder}{%  
8921 \csuse{@glscompstyle@longragged}%  
8922 }%
```

Backward compatible longragged3col style.

```
8923 \compatglossarystyle{longragged3col}{%  
8924 \renewcommand*{\glossaryentryfield}[5]{%  
8925 \glstryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\tabularnewline}%  
8926 \renewcommand*{\glossarysubentryfield}[6]{%
```

```

8927      &
8928      \glssubentryitem{##2}%
8929      \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%
8930 }%

Backward compatible longragged3colborder style.
8931 \compatglossarystyle{longragged3colborder}{%
8932 \csuse{@glscmpstyle@longragged3col}%
8933 }%

Backward compatible longragged3colheader style.
8934 \compatglossarystyle{longragged3colheader}{%
8935 \csuse{@glscmpstyle@longragged3col}%
8936 }%

Backward compatible longragged3colheaderborder style.
8937 \compatglossarystyle{longragged3colheaderborder}{%
8938 \csuse{@glscmpstyle@longragged3col}%
8939 }%

Backward compatible altlongragged4col style.
8940 \compatglossarystyle{altlongragged4col}{%
8941 \renewcommand*{\glossaryentryfield}[5]{%
8942 \glssentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\tabularnewline}%
8943 \renewcommand*{\glossarysubentryfield}[6]{%
8944 &
8945 \glssubentryitem{##2}%
8946 \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%
8947 }%

Backward compatible altlongragged4colheader style.
8948 \compatglossarystyle{altlongragged4colheader}{%
8949 \csuse{@glscmpstyle@altlong4col}%
8950 }%

Backward compatible altlongragged4colborder style.
8951 \compatglossarystyle{altlongragged4colborder}{%
8952 \csuse{@glscmpstyle@altlong4col}%
8953 }%

Backward compatible altlongragged4colheaderborder style.
8954 \compatglossarystyle{altlongragged4colheaderborder}{%
8955 \csuse{@glscmpstyle@altlong4col}%
8956 }%

Backward compatible index style.
8957 \compatglossarystyle{index}{%
8958 \renewcommand*{\glossaryentryfield}[5]{%
8959 \item\glssentryitem{##1}\textbf{\glstarget{##1}{##2}}%
8960 \ifx\relax##4\relax
8961 \else
8962 \space(##4)%

```

```

8963     \fi
8964     \space ##3\glspostdescription \space ##5}%
8965 \renewcommand*\glossarysubentryfield}[6]{%
8966     \ifcase##1\relax
8967     % level 0
8968     \item
8969     \or
8970     % level 1
8971     \subitem
8972     \glssubentryitem{##2}%
8973     \else
8974     % all other levels
8975     \subsubitem
8976     \fi
8977     \textbf{\glstarget{##2}{##3}}%
8978     \ifx\relax##5\relax
8979     \else
8980     \space(##5)%
8981     \fi
8982     \space##4\glspostdescription\space ##6}%
8983 }%

```

Backward compatible indexgroup style.

```

8984 \compatglossarystyle{indexgroup}{%
8985 \csuse{@glscmpstyle@index}%
8986 }%

```

Backward compatible indexhypergroup style.

```

8987 \compatglossarystyle{indexhypergroup}{%
8988 \csuse{@glscmpstyle@index}%
8989 }%

```

Backward compatible tree style.

```

8990 \compatglossarystyle{tree}{%
8991 \renewcommand*\glossaryentryfield}[5]{%
8992     \hangindent0pt\relax
8993     \parindent0pt\relax
8994     \glstentryitem{##1}\textbf{\glstarget{##1}{##2}}%
8995     \ifx\relax##4\relax
8996     \else
8997     \space(##4)%
8998     \fi
8999     \space ##3\glspostdescription \space ##5\par}%
9000 \renewcommand*\glossarysubentryfield}[6]{%
9001     \hangindent##1\glstreeindent\relax
9002     \parindent##1\glstreeindent\relax
9003     \ifnum##1=1\relax
9004     \glssubentryitem{##2}%
9005     \fi
9006     \textbf{\glstarget{##2}{##3}}%
9007     \ifx\relax##5\relax

```

```

9008     \else
9009         \space{##5}%
9010     \fi
9011     \space##4\glspostdescription\space ##6\par}%
9012 }%

    Backward compatible treegroup style.
9013 \compatglossarystyle{treegroup}{%
9014     \csuse{@glscmpstyle@tree}%
9015 }%

    Backward compatible treehypergroup style.
9016 \compatglossarystyle{treehypergroup}{%
9017     \csuse{@glscmpstyle@tree}%
9018 }%

    Backward compatible treenoname style.
9019 \compatglossarystyle{treenoname}{%
9020     \renewcommand{\glossaryentryfield}[5]{%
9021         \hangindent0pt\relax
9022         \parindent0pt\relax
9023         \glstryitem{##1}\textbf{\glstarget{##1}{##2}}%
9024         \ifx\relax##4\relax
9025             \else
9026                 \space{##4}%
9027             \fi
9028         \space ##3\glspostdescription \space ##5\par}%
9029     \renewcommand{\glossarysubentryfield}[6]{%
9030         \hangindent##1\glstreeindent\relax
9031         \parindent##1\glstreeindent\relax
9032         \ifnum##1=1\relax
9033             \glssubentryitem{##2}%
9034         \fi
9035         \glstarget{##2}{\strut}%
9036         ##4\glspostdescription\space ##6\par}%
9037 }%

    Backward compatible treenonamegroup style.
9038 \compatglossarystyle{treenonamegroup}{%
9039     \csuse{@glscmpstyle@treenoname}%
9040 }%

    Backward compatible treenonamehypergroup style.
9041 \compatglossarystyle{treenonamehypergroup}{%
9042     \csuse{@glscmpstyle@treenoname}%
9043 }%

    Backward compatible alttree style.
9044 \compatglossarystyle{alttree}{%
9045     \renewcommand{\glossaryentryfield}[5]{%
9046         \ifnum \@gls@prevlevel=0\relax
9047         \else

```

```

9048     \settowidth{\glstreeindent}{\textbf{\@glswidestname\space}}%
9049     \hangindent\glstreeindent
9050     \parindent\glstreeindent
9051     \fi
9052     \makebox[0pt][r]{\makebox[\glstreeindent][l]{%
9053       \glstentryitem{##1}\textbf{\glstarget{##1}{##2}}}%
9054     \ifx\relax##4\relax
9055     \else
9056       (##4)\space
9057     \fi
9058     ##3\glspostdescription \space ##5\par
9059     \def\@gls@prevlevel{0}%
9060 }%
9061 \renewcommand{\glossarysubentryfield}[6]{%
9062   \ifnum##1=1\relax
9063     \glssubentryitem{##2}%
9064   \fi
9065   \ifnum\@gls@prevlevel=##1\relax
9066   \else
9067     \@ifundefined{\@glswidestname\romannumeral##1}{%
9068       \settowidth{\gls@tmplen}{\textbf{\@glswidestname\space}}{%
9069       \settowidth{\gls@tmplen}{\textbf{%
9070         \csname \@glswidestname\romannumeral##1\endcsname\space}}}%
9071     \ifnum\@gls@prevlevel<##1\relax
9072       \setlength\glstreeindent\gls@tmplen
9073       \addtolength\glstreeindent\parindent
9074       \parindent\glstreeindent
9075     \else
9076       \@ifundefined{\@glswidestname\romannumeral\@gls@prevlevel}{%
9077         \settowidth{\glstreeindent}{\textbf{%
9078           \@glswidestname\space}}{%
9079         \settowidth{\glstreeindent}{\textbf{%
9080           \csname \@glswidestname\romannumeral\@gls@prevlevel
9081             \endcsname\space}}}%
9082       \addtolength\parindent{-\glstreeindent}%
9083       \setlength\glstreeindent\parindent
9084     \fi
9085   \fi
9086   \hangindent\glstreeindent
9087   \makebox[0pt][r]{\makebox[\gls@tmplen][l]{%
9088     \textbf{\glstarget{##2}{##3}}}%
9089   \ifx##5\relax\relax
9090   \else
9091     (##5)\space
9092   \fi
9093   ##4\glspostdescription\space ##6\par
9094   \def\@gls@prevlevel{##1}%
9095 }%
9096 }%

```

Backward compatible alttreegroup style.

```
9097 \compatglossarystyle{alttreegroup}{%  
9098 \csuse{@glscompstyle@alttree}%  
9099 }%
```

Backward compatible alttreehypergroup style.

```
9100 \compatglossarystyle{alttreehypergroup}{%  
9101 \csuse{@glscompstyle@alttree}%  
9102 }%
```

Backward compatible mcolindex style.

```
9103 \compatglossarystyle{mcolindex}{%  
9104 \csuse{@glscompstyle@index}%  
9105 }%
```

Backward compatible mcolindexgroup style.

```
9106 \compatglossarystyle{mcolindexgroup}{%  
9107 \csuse{@glscompstyle@index}%  
9108 }%
```

Backward compatible mcolindexhypergroup style.

```
9109 \compatglossarystyle{mcolindexhypergroup}{%  
9110 \csuse{@glscompstyle@index}%  
9111 }%
```

Backward compatible mcoltree style.

```
9112 \compatglossarystyle{mcoltree}{%  
9113 \csuse{@glscompstyle@tree}%  
9114 }%
```

Backward compatible mcoltreegroup style.

```
9115 \compatglossarystyle{mcolindextreegroup}{%  
9116 \csuse{@glscompstyle@tree}%  
9117 }%
```

Backward compatible mcoltreehypergroup style.

```
9118 \compatglossarystyle{mcolindextreehypergroup}{%  
9119 \csuse{@glscompstyle@tree}%  
9120 }%
```

Backward compatible mcoltreename style.

```
9121 \compatglossarystyle{mcoltreename}{%  
9122 \csuse{@glscompstyle@tree}%  
9123 }%
```

Backward compatible mcoltreenamegroup style.

```
9124 \compatglossarystyle{mcoltreenamegroup}{%  
9125 \csuse{@glscompstyle@tree}%  
9126 }%
```

Backward compatible mcoltreenamehypergroup style.

```
9127 \compatglossarystyle{mcoltreenamehypergroup}{%  
9128 \csuse{@glscompstyle@tree}%  
9129 }%
```

Backward compatible mcolalmtree style.

```
9130 \compatglossarystyle{mcolalmtree}{%
9131 \csuse{@glscmpstyle@almtree}%
9132 }%
```

Backward compatible mcolalmtreegroup style.

```
9133 \compatglossarystyle{mcolalmtreegroup}{%
9134 \csuse{@glscmpstyle@almtree}%
9135 }%
```

Backward compatible mcolalmtreehypergroup style.

```
9136 \compatglossarystyle{mcolalmtreehypergroup}{%
9137 \csuse{@glscmpstyle@almtree}%
9138 }%
```

Backward compatible superragged style.

```
9139 \compatglossarystyle{superragged}{%
9140 \renewcommand*{\glossaryentryfield}[5]{%
9141 \glstryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%
9142 \tabularnewline}%
9143 \renewcommand*{\glossarysubentryfield}[6]{%
9144 &
9145 \glssubentryitem{##2}%
9146 \glstarget{##2}{\strut}##4\glspostdescription\space ##6%
9147 \tabularnewline}%
9148 }%
```

Backward compatible superraggedborder style.

```
9149 \compatglossarystyle{superraggedborder}{%
9150 \csuse{@glscmpstyle@superragged}%
9151 }%
```

Backward compatible superraggedheader style.

```
9152 \compatglossarystyle{superraggedheader}{%
9153 \csuse{@glscmpstyle@superragged}%
9154 }%
```

Backward compatible superraggedheaderborder style.

```
9155 \compatglossarystyle{superraggedheaderborder}{%
9156 \csuse{@glscmpstyle@superragged}%
9157 }%
```

Backward compatible superragged3col style.

```
9158 \compatglossarystyle{superragged3col}{%
9159 \renewcommand*{\glossaryentryfield}[5]{%
9160 \glstryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\tabularnewline}%
9161 \renewcommand*{\glossarysubentryfield}[6]{%
9162 &
9163 \glssubentryitem{##2}%
9164 \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%
9165 }%
```



Backward compatible superragged3colborder style.

```
9166 \compatglossarystyle{superragged3colborder}{%
9167 \csuse{@glscompstyle@superragged3col}%
9168 }%
```

Backward compatible superragged3colheader style.

```
9169 \compatglossarystyle{superragged3colheader}{%
9170 \csuse{@glscompstyle@superragged3col}%
9171 }%
```

Backward compatible superragged3colheaderborder style.

```
9172 \compatglossarystyle{superragged3colheaderborder}{%
9173 \csuse{@glscompstyle@superragged3col}%
9174 }%
```

Backward compatible altsuperragged4col style.

```
9175 \compatglossarystyle{altsuperragged4col}{%
9176 \renewcommand*{\glossaryentryfield}[5]{%
9177 \glstarget{##1}{\glstarget{##1}{##2} & ##3 & ##4 & ##5\tabularnewline}%
9178 \renewcommand*{\glossarysubentryfield}[6]{%
9179 &
9180 \glssubentryitem{##2}%
9181 \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%
9182 }%
```

Backward compatible altsuperragged4colheader style.

```
9183 \compatglossarystyle{altsuperragged4colheader}{%
9184 \csuse{@glscompstyle@altsuperragged4col}%
9185 }%
```

Backward compatible altsuperragged4colborder style.

```
9186 \compatglossarystyle{altsuperragged4colborder}{%
9187 \csuse{@glscompstyle@altsuperragged4col}%
9188 }%
```

Backward compatible altsuperragged4colheaderborder style.

```
9189 \compatglossarystyle{altsuperragged4colheaderborder}{%
9190 \csuse{@glscompstyle@altsuperragged4col}%
9191 }%
```

Backward compatible super style.

```
9192 \compatglossarystyle{super}{%
9193 \renewcommand*{\glossaryentryfield}[5]{%
9194 \glstarget{##1}{\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\}%
9195 \renewcommand*{\glossarysubentryfield}[6]{%
9196 &
9197 \glssubentryitem{##2}%
9198 \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\}%
9199 }%
```

Backward compatible superborder style.

```
9200 \compatglossarystyle{superborder}{%

```

```
9201 \csuse{@glscompstyle@super}%
9202 }%
```

Backward compatible superheader style.

```
9203 \compatglossarystyle{superheader}{%
9204 \csuse{@glscompstyle@super}%
9205 }%
```

Backward compatible superheaderborder style.

```
9206 \compatglossarystyle{superheaderborder}{%
9207 \csuse{@glscompstyle@super}%
9208 }%
```

Backward compatible super3col style.

```
9209 \compatglossarystyle{super3col}{%
9210 \renewcommand*{\glossaryentryfield}[5]{%
9211 \glstentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\\}%
9212 \renewcommand*{\glossarysubentryfield}[6]{%
9213 &
9214 \glssubentryitem{##2}%
9215 \glstarget{##2}{\strut}##4 & ##6\\}%
9216 }%
```

Backward compatible super3colborder style.

```
9217 \compatglossarystyle{super3colborder}{%
9218 \csuse{@glscompstyle@super3col}%
9219 }%
```

Backward compatible super3colheader style.

```
9220 \compatglossarystyle{super3colheader}{%
9221 \csuse{@glscompstyle@super3col}%
9222 }%
```

Backward compatible super3colheaderborder style.

```
9223 \compatglossarystyle{super3colheaderborder}{%
9224 \csuse{@glscompstyle@super3col}%
9225 }%
```

Backward compatible super4col style.

```
9226 \compatglossarystyle{super4col}{%
9227 \renewcommand*{\glossaryentryfield}[5]{%
9228 \glstentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\\}%
9229 \renewcommand*{\glossarysubentryfield}[6]{%
9230 &
9231 \glssubentryitem{##2}%
9232 \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%
9233 }%
```

Backward compatible super4colheader style.

```
9234 \compatglossarystyle{super4colheader}{%
9235 \csuse{@glscompstyle@super4col}%
9236 }%
```

Backward compatible super4colborder style.

```
9237 \compatglossarystyle{super4colborder}{%  
9238 \csuse{@glscompstyle@super4col}%  
9239 }%
```

Backward compatible super4colheaderborder style.

```
9240 \compatglossarystyle{super4colheaderborder}{%  
9241 \csuse{@glscompstyle@super4col}%  
9242 }%
```

Backward compatible altsuper4col style.

```
9243 \compatglossarystyle{altsuper4col}{%  
9244 \csuse{@glscompstyle@super4col}%  
9245 }%
```

Backward compatible altsuper4colheader style.

```
9246 \compatglossarystyle{altsuper4colheader}{%  
9247 \csuse{@glscompstyle@super4col}%  
9248 }%
```

Backward compatible altsuper4colborder style.

```
9249 \compatglossarystyle{altsuper4colborder}{%  
9250 \csuse{@glscompstyle@super4col}%  
9251 }%
```

Backward compatible altsuper4colheaderborder style.

```
9252 \compatglossarystyle{altsuper4colheaderborder}{%  
9253 \csuse{@glscompstyle@super4col}%  
9254 }%
```

## 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.

```
9255 \NeedsTeXFormat{LaTeX2e}
```

Package version number now in line with main glossaries package number but will only be updated when glossaries-accsupp.sty is modified.

```
9256 \ProvidesPackage{glossaries-accsupp}[2014/07/30 v4.08 (NLCT)]  
9257 Experimental glossaries accessibility
```

Pass all options to glossaries:

```
9258 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{glossaries}}
```

Process options:

```
9259 \ProcessOptions
```

`\compatibleglossentry` Override style compatibility macros:

```
9260 \def\compatibleglossentry#1#2{%
```

```

9261 \toks@{#2}%
9262 \protected@edef\@do@glossentry{%
9263   \noexpand\accsuppglossaryentryfield{#1}%
9264   {\noexpand\glsnamefont
9265     {\expandafter\expandonce\csname glo@\glsdetoklabel{#1}@name\endcsname}}}%
9266   {\expandafter\expandonce\csname glo@\glsdetoklabel{#1}@desc\endcsname}}}%
9267   {\expandafter\expandonce\csname glo@\glsdetoklabel{#1}@symbol\endcsname}}}%
9268   {\the\toks@}%
9269 }%
9270 \@do@glossentry
9271 }

```

atiblessubglossentry

```

9272 \def\compatiblessubglossentry#1#2#3{%
9273   \toks@{#3}%
9274   \protected@edef\@do@subglossentry{%
9275     \noexpand\accsuppglossarysubentryfield{\number#1}%
9276     {#2}%
9277     {\noexpand\glsnamefont
9278       {\expandafter\expandonce\csname glo@\glsdetoklabel{#2}@name\endcsname}}}%
9279     {\expandafter\expandonce\csname glo@\glsdetoklabel{#2}@desc\endcsname}}}%
9280     {\expandafter\expandonce\csname glo@\glsdetoklabel{#2}@symbol\endcsname}}}%
9281     {\the\toks@}%
9282   }%
9283   \@do@subglossentry
9284 }

```

Required packages:

```

9285 \RequirePackage{glossaries}
9286 \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:

```

9287 \define@key{glossentry}{access}{%
9288   \def\@glo@access{#1}%
9289 }

```

**textaccess** The replacement text corresponding to the text key:

```

9290 \define@key{glossentry}{textaccess}{%
9291   \def\@glo@textaccess{#1}%
9292 }

```

firstaccess The replacement text corresponding to the first key:

```
9293 \define@key{glossentry}{firstaccess}{%
9294   \def\@glo@firstaccess{#1}%
9295 }
```

pluralaccess The replacement text corresponding to the plural key:

```
9296 \define@key{glossentry}{pluralaccess}{%
9297   \def\@glo@pluralaccess{#1}%
9298 }
```

firstpluralaccess The replacement text corresponding to the firstplural key:

```
9299 \define@key{glossentry}{firstpluralaccess}{%
9300   \def\@glo@firstpluralaccess{#1}%
9301 }
```

symbolaccess The replacement text corresponding to the symbol key:

```
9302 \define@key{glossentry}{symbolaccess}{%
9303   \def\@glo@symbolaccess{#1}%
9304 }
```

symbolpluralaccess The replacement text corresponding to the symbolplural key:

```
9305 \define@key{glossentry}{symbolpluralaccess}{%
9306   \def\@glo@symbolpluralaccess{#1}%
9307 }
```

descriptionaccess The replacement text corresponding to the description key:

```
9308 \define@key{glossentry}{descriptionaccess}{%
9309   \def\@glo@descaccess{#1}%
9310 }
```

descriptionpluralaccess The replacement text corresponding to the descriptionplural key:

```
9311 \define@key{glossentry}{descriptionpluralaccess}{%
9312   \def\@glo@descpluralaccess{#1}%
9313 }
```

shortaccess The replacement text corresponding to the short key:

```
9314 \define@key{glossentry}{shortaccess}{%
9315   \def\@glo@shortaccess{#1}%
9316 }
```

shortpluralaccess The replacement text corresponding to the shortplural key:

```
9317 \define@key{glossentry}{shortpluralaccess}{%
9318   \def\@glo@shortpluralaccess{#1}%
9319 }
```

longaccess The replacement text corresponding to the long key:

```
9320 \define@key{glossentry}{longaccess}{%
9321   \def\@glo@longaccess{#1}%
9322 }
```

longpluralaccess The replacement text corresponding to the longplural key:

```
9323 \define@key{glossentry}{longpluralaccess}{%
9324   \def\@glo@longpluralaccess{#1}%
9325 }
```

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:

```
9326 \appto\@gls@keymap{,%
9327   {access}{access},%
9328   {textaccess}{textaccess},%
9329   {firstaccess}{firstaccess},%
9330   {pluralaccess}{pluralaccess},%
9331   {firstpluralaccess}{firstpluralaccess},%
9332   {symbolaccess}{symbolaccess},%
9333   {symbolpluralaccess}{symbolpluralaccess},%
9334   {descaccess}{descaccess},%
9335   {descpluralaccess}{descpluralaccess},%
9336   {shortaccess}{shortaccess},%
9337   {shortpluralaccess}{shortpluralaccess},%
9338   {longaccess}{longaccess},%
9339   {longpluralaccess}{longpluralaccess}%
9340 }
```

\@gls@noaccess Indicates that no replacement text has been provided.

```
9341 \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):

```
9342 \let\@gls@oldnewglossaryentryprehook\@newglossaryentryprehook
9343 \renewcommand*{\@newglossaryentryprehook}{%
9344   \@gls@oldnewglossaryentryprehook
9345   \def\@glo@access{\@glo@symbol}%

```

Initialise the other keys:

```
9346   \def\@glo@textaccess{\@glo@access}%
9347   \def\@glo@firstaccess{\@glo@access}%
9348   \def\@glo@pluralaccess{\@glo@textaccess}%
9349   \def\@glo@firstpluralaccess{\@glo@pluralaccess}%
9350   \def\@glo@symbolaccess{\relax}%
9351   \def\@glo@symbolpluralaccess{\@glo@symbolaccess}%
9352   \def\@glo@descaccess{\relax}%
9353   \def\@glo@descpluralaccess{\@glo@descaccess}%
9354   \def\@glo@shortaccess{\relax}%
9355   \def\@glo@shortpluralaccess{\@glo@shortaccess}%
9356   \def\@glo@longaccess{\relax}%
9357   \def\@glo@longpluralaccess{\@glo@longaccess}%
9358 }
```

Add to the end hook:

```
9359 \let\@gls@oldnewglossaryentryposthook\@newglossaryentryposthook
9360 \renewcommand*{\@newglossaryentryposthook}{%
9361   \@gls@oldnewglossaryentryposthook
```

Store the access information:

```
9362   \expandafter
9363     \protected@xdef\csname glo@\@glo@label @access\endcsname{%
9364       \@glo@access}%
9365   \expandafter
9366     \protected@xdef\csname glo@\@glo@label @textaccess\endcsname{%
9367       \@glo@textaccess}%
9368   \expandafter
9369     \protected@xdef\csname glo@\@glo@label @firstaccess\endcsname{%
9370       \@glo@firstaccess}%
9371   \expandafter
9372     \protected@xdef\csname glo@\@glo@label @pluralaccess\endcsname{%
9373       \@glo@pluralaccess}%
9374   \expandafter
9375     \protected@xdef\csname glo@\@glo@label @firstpluralaccess\endcsname{%
9376       \@glo@firstpluralaccess}%
9377   \expandafter
9378     \protected@xdef\csname glo@\@glo@label @symbolaccess\endcsname{%
9379       \@glo@symbolaccess}%
9380   \expandafter
9381     \protected@xdef\csname glo@\@glo@label @symbolpluralaccess\endcsname{%
9382       \@glo@symbolpluralaccess}%
9383   \expandafter
9384     \protected@xdef\csname glo@\@glo@label @descaccess\endcsname{%
9385       \@glo@descaccess}%
9386   \expandafter
9387     \protected@xdef\csname glo@\@glo@label @descpluralaccess\endcsname{%
9388       \@glo@descpluralaccess}%
9389   \expandafter
9390     \protected@xdef\csname glo@\@glo@label @shortaccess\endcsname{%
9391       \@glo@shortaccess}%
9392   \expandafter
9393     \protected@xdef\csname glo@\@glo@label @shortpluralaccess\endcsname{%
9394       \@glo@shortpluralaccess}%
9395   \expandafter
9396     \protected@xdef\csname glo@\@glo@label @longaccess\endcsname{%
9397       \@glo@longaccess}%
9398   \expandafter
9399     \protected@xdef\csname glo@\@glo@label @longpluralaccess\endcsname{%
9400       \@glo@longpluralaccess}%
9401 }
```

## 7.2 Accessing Replacement Text

`\glsentryaccess` Get the value of the access key for the entry with the given label:

```
9402 \newcommand*{\glsentryaccess}[1]{%
9403   \@gls@entry@field{#1}{access}%
9404 }
```

`\glsentrytextaccess` Get the value of the textaccess key for the entry with the given label:

```
9405 \newcommand*{\glsentrytextaccess}[1]{%
9406   \@gls@entry@field{#1}{textaccess}%
9407 }
```

`\glsentryfirstaccess` Get the value of the firstaccess key for the entry with the given label:

```
9408 \newcommand*{\glsentryfirstaccess}[1]{%
9409   \@gls@entry@field{#1}{firstaccess}%
9410 }
```

`\glsentrypluralaccess` Get the value of the pluralaccess key for the entry with the given label:

```
9411 \newcommand*{\glsentrypluralaccess}[1]{%
9412   \@gls@entry@field{#1}{pluralaccess}%
9413 }
```

`\glsentryfirstpluralaccess` Get the value of the firstpluralaccess key for the entry with the given label:

```
9414 \newcommand*{\glsentryfirstpluralaccess}[1]{%
9415   \csname glo@#1@firstpluralaccess\endcsname
9416 }
```

`\glsentrysymbolaccess` Get the value of the symbolaccess key for the entry with the given label:

```
9417 \newcommand*{\glsentrysymbolaccess}[1]{%
9418   \@gls@entry@field{#1}{symbolaccess}%
9419 }
```

`\glsentrysymbolpluralaccess` Get the value of the symbolpluralaccess key for the entry with the given label:

```
9420 \newcommand*{\glsentrysymbolpluralaccess}[1]{%
9421   \@gls@entry@field{#1}{symbolpluralaccess}%
9422 }
```

`\glsentrydescaccess` Get the value of the descriptionaccess key for the entry with the given label:

```
9423 \newcommand*{\glsentrydescaccess}[1]{%
9424   \@gls@entry@field{#1}{descaccess}%
9425 }
```

`\glsentrydescpluralaccess` Get the value of the descriptionpluralaccess key for the entry with the given label:

```
9426 \newcommand*{\glsentrydescpluralaccess}[1]{%
9427   \@gls@entry@field{#1}{descaccess}%
9428 }
```



`\glsentryshortaccess` Get the value of the shortaccess key for the entry with the given label:

```
9429 \newcommand*{\glsentryshortaccess}[1]{%
9430   \@gls@entry@field{#1}{shortaccess}%
9431 }
```

`\glsentryshortpluralaccess` Get the value of the shortpluralaccess key for the entry with the given label:

```
9432 \newcommand*{\glsentryshortpluralaccess}[1]{%
9433   \@gls@entry@field{#1}{shortpluralaccess}%
9434 }
```

`\glsentrylongaccess` Get the value of the longaccess key for the entry with the given label:

```
9435 \newcommand*{\glsentrylongaccess}[1]{%
9436   \@gls@entry@field{#1}{longaccess}%
9437 }
```

`\glsentrylongpluralaccess` Get the value of the longpluralaccess key for the entry with the given label:

```
9438 \newcommand*{\glsentrylongpluralaccess}[1]{%
9439   \@gls@entry@field{#1}{longpluralaccess}%
9440 }
```

`\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.)

```
9441 \newcommand*{\glsaccsupp}[2]{%
9442   \BeginAccSupp{ActualText=#1}#2\EndAccSupp{}%
9443 }
```

`\xglsaccsupp` Fully expands replacement text before calling `\glsaccsupp`

```
9444 \newcommand*{\xglsaccsupp}[2]{%
9445   \protected@edef\@gls@replacementtext{#1}%
9446   \expandafter\glsaccsupp\expandafter{\@gls@replacementtext}{#2}%
9447 }
```

`@gls@access@display`

```
9448 \newcommand*{\@gls@access@display}[2]{%
9449   \protected@edef\@glo@access{#2}%
9450   \ifx\@glo@access\@gls@noaccess
9451     #1%
9452   \else
9453     \xglsaccsupp{\@glo@access}{#1}%
9454   \fi
9455 }
```

`\glsnameaccessdisplay` Displays the first argument with the accessibility text for the entry with the label given by the second argument (if set).

```
9456 \DeclareRobustCommand*{\glsnameaccessdisplay}[2]{%
9457   \@gls@access@display{#1}{\glsentryaccess{#2}}%
9458 }
```

lstextaccessdisplay As above but for the textaccess replacement text.

```

9459 \DeclareRobustCommand*\glstextaccessdisplay}[2]{%
9460   \@gls@access@display{#1}{\glsentrytextaccess{#2}}}%
9461 }

```

pluralaccessdisplay As above but for the pluralaccess replacement text.

```

9462 \DeclareRobustCommand*\glspluralaccessdisplay}[2]{%
9463   \@gls@access@display{#1}{\glsentrypluralaccess{#2}}}%
9464 }

```

sfirstaccessdisplay As above but for the firstaccess replacement text.

```

9465 \DeclareRobustCommand*\glsfirstaccessdisplay}[2]{%
9466   \@gls@access@display{#1}{\glsentryfirstaccess{#2}}}%
9467 }

```

pluralaccessdisplay As above but for the firstpluralaccess replacement text.

```

9468 \DeclareRobustCommand*\glsfirstpluralaccessdisplay}[2]{%
9469   \@gls@access@display{#1}{\glsentryfirstpluralaccess{#2}}}%
9470 }

```

symbolaccessdisplay As above but for the symbolaccess replacement text.

```

9471 \DeclareRobustCommand*\glssymbolaccessdisplay}[2]{%
9472   \@gls@access@display{#1}{\glsentrysymbolaccess{#2}}}%
9473 }

```

pluralaccessdisplay As above but for the symbolpluralaccess replacement text.

```

9474 \DeclareRobustCommand*\glssymbolpluralaccessdisplay}[2]{%
9475   \@gls@access@display{#1}{\glsentrysymbolpluralaccess{#2}}}%
9476 }

```

ptionaccessdisplay As above but for the descriptionaccess replacement text.

```

9477 \DeclareRobustCommand*\glsdescriptionaccessdisplay}[2]{%
9478   \@gls@access@display{#1}{\glsentrydescaccess{#2}}}%
9479 }

```

pluralaccessdisplay As above but for the descriptionpluralaccess replacement text.

```

9480 \DeclareRobustCommand*\glsdescriptionpluralaccessdisplay}[2]{%
9481   \@gls@access@display{#1}{\glsentrydescpluralaccess{#2}}}%
9482 }

```

sshortaccessdisplay As above but for the shortaccess replacement text.

```

9483 \DeclareRobustCommand*\glsshortaccessdisplay}[2]{%
9484   \@gls@access@display{#1}{\glsentryshortaccess{#2}}}%
9485 }

```

pluralaccessdisplay As above but for the shortpluralaccess replacement text.

```

9486 \DeclareRobustCommand*\glsshortpluralaccessdisplay}[2]{%
9487   \@gls@access@display{#1}{\glsentryshortpluralaccess{#2}}}%
9488 }

```

`longaccessdisplay` As above but for the `longaccess` replacement text.

```

9489 \DeclareRobustCommand*\glslongaccessdisplay}[2]{%
9490   \@gls@access@display{#1}{\glsentrylongaccess{#2}}}%
9491 }

```

`pluralaccessdisplay` As above but for the `longpluralaccess` replacement text.

```

9492 \DeclareRobustCommand*\glslongpluralaccessdisplay}[2]{%
9493   \@gls@access@display{#1}{\glsentrylongpluralaccess{#2}}}%
9494 }

```

`\glsaccessdisplay` Gets the replacement text corresponding to the named key given by the first argument and calls the appropriate command defined above.

```

9495 \DeclareRobustCommand*\glsaccessdisplay}[3]{%
9496   \@ifundefined{gls#1accessdisplay}%
9497   {%
9498     \PackageError{glossaries-accsupp}{No accessibility support
9499     for key ‘#1’}{}%
9500   }%
9501   {%
9502     \csname gls#1accessdisplay\endcsname{#2}{#3}%
9503   }%
9504 }

```

`ls@default@entryfmt` Redefine the default entry format to use accessibility information

```

9505 \renewcommand*\@@gls@default@entryfmt}[2]{%
9506   \ifdefempty\glscustomtext
9507   {%
9508     \glsifplural
9509     {%

```

Plural form

```

9510       \glscapscase
9511       {%

```

Don't adjust case

```

9512       \ifglsused\glslabel
9513       {%

```

Subsequent use

```

9514         #2{\glspluralaccessdisplay
9515           {\glsentryplural{\glslabel}}{\glslabel}}%
9516         {\glsdescriptionpluralaccessdisplay
9517           {\glsentrydescplural{\glslabel}}{\glslabel}}%
9518         {\glssymbolpluralaccessdisplay
9519           {\glsentrysymbolplural{\glslabel}}{\glslabel}}
9520         {\glsinsert}%
9521       }%
9522     }%

```

First use

```

9523      #1{\glsfirstpluralaccessdisplay
9524          {\glsentryfirstplural{\glslabel}}{\glslabel}}%
9525          {\glsdescriptionpluralaccessdisplay
9526              {\glsentrydescplural{\glslabel}}{\glslabel}}%
9527          {\glssymbolpluralaccessdisplay
9528              {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
9529          {\glsinsert}}%
9530      }%
9531  }%
9532  {%

```

Make first letter upper case

```

9533      \ifglsused\glslabel
9534      {%

```

Subsequent use.

```

9535      #2{\glspluralaccessdisplay
9536          {\Glsentryplural{\glslabel}}{\glslabel}}%
9537          {\glsdescriptionpluralaccessdisplay
9538              {\glsentrydescplural{\glslabel}}{\glslabel}}%
9539          {\glssymbolpluralaccessdisplay
9540              {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
9541          {\glsinsert}}%
9542      }%
9543  {%

```

First use

```

9544      #1{\glsfirstpluralaccessdisplay
9545          {\Glsentryfirstplural{\glslabel}}{\glslabel}}%
9546          {\glsdescriptionpluralaccessdisplay
9547              {\glsentrydescplural{\glslabel}}{\glslabel}}%
9548          {\glssymbolpluralaccessdisplay
9549              {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
9550          {\glsinsert}}%
9551      }%
9552  }%
9553  {%

```

Make all upper case

```

9554      \ifglsused\glslabel
9555      {%

```

Subsequent use

```

9556      \MakeUppercase{%
9557          #2{\glspluralaccessdisplay
9558              {\glsentryplural{\glslabel}}{\glslabel}}%
9559              {\glsdescriptionpluralaccessdisplay
9560                  {\glsentrydescplural{\glslabel}}{\glslabel}}%
9561              {\glssymbolpluralaccessdisplay
9562                  {\glsentrysymbolplural{\glslabel}}{\glslabel}}%

```

```

9563         {\glsinsert}}}%
9564     }%
9565     {%

```

#### First use

```

9566     \MakeUppercase{%
9567         #1{\glsfirstpluralaccessdisplay
9568             {\glsentryfirstplural{\glslabel}}{\glslabel}}}%
9569         {\glsdescriptionpluralaccessdisplay
9570             {\glsentrydescplural{\glslabel}}{\glslabel}}}%
9571         {\glsymbolpluralaccessdisplay
9572             {\glsentrysymbolplural{\glslabel}}{\glslabel}}}%
9573         {\glsinsert}}}%
9574     }%
9575 }%
9576 }%
9577 {%

```

#### Singular form

```

9578     \glscapscase
9579     {%

```

#### Don't adjust case

```

9580     \ifglsused\glslabel
9581     {%

```

#### Subsequent use

```

9582     #2{\glstextaccessdisplay
9583         {\glsentrytext{\glslabel}}{\glslabel}}}%
9584         {\glsdescriptionaccessdisplay
9585             {\glsentrydesc{\glslabel}}{\glslabel}}}%
9586         {\glsymbolaccessdisplay
9587             {\glsentrysymbol{\glslabel}}{\glslabel}}}%
9588         {\glsinsert}}}%
9589     }%
9590     {%

```

#### First use

```

9591     #1{\glsfirstaccessdisplay
9592         {\glsentryfirst{\glslabel}}{\glslabel}}}%
9593         {\glsdescriptionaccessdisplay
9594             {\glsentrydesc{\glslabel}}{\glslabel}}}%
9595         {\glsymbolaccessdisplay
9596             {\glsentrysymbol{\glslabel}}{\glslabel}}}%
9597         {\glsinsert}}}%
9598     }%
9599     }%
9600     {%

```

#### Make first letter upper case

```

9601     \ifglsused\glslabel
9602     {%

```

### Subsequent use

```

9603      #2{\glstextaccessdisplay
9604          {\Glentrytext{\glslabel}}{\glslabel}}%
9605          {\glsdescriptionaccessdisplay
9606              {\glentrydesc{\glslabel}}{\glslabel}}%
9607          {\glssymbolaccessdisplay
9608              {\glentrysymbol{\glslabel}}{\glslabel}}%
9609          {\glsinsert}}%
9610      }%
9611      {%

```

### First use

```

9612      #1{\glsfirstaccessdisplay
9613          {\Glentryfirst{\glslabel}}{\glslabel}}%
9614          {\glsdescriptionaccessdisplay
9615              {\glentrydesc{\glslabel}}{\glslabel}}%
9616          {\glssymbolaccessdisplay
9617              {\glentrysymbol{\glslabel}}{\glslabel}}%
9618          {\glsinsert}}%
9619      }%
9620      }%
9621      {%

```

### Make all upper case

```

9622      \ifglsused\glslabel
9623      {%

```

### Subsequent use

```

9624      \MakeUppercase{%
9625      #2{\glstextaccessdisplay
9626          {\glentrytext{\glslabel}}{\glslabel}}%
9627          {\glsdescriptionaccessdisplay
9628              {\glentrydesc{\glslabel}}{\glslabel}}%
9629          {\glssymbolaccessdisplay
9630              {\glentrysymbol{\glslabel}}{\glslabel}}%
9631          {\glsinsert}}%
9632      }%
9633      {%

```

### First use

```

9634      \MakeUppercase{%
9635      #1{\glsfirstaccessdisplay
9636          {\glentryfirst{\glslabel}}{\glslabel}}%
9637          {\glsdescriptionaccessdisplay
9638              {\glentrydesc{\glslabel}}{\glslabel}}%
9639          {\glssymbolaccessdisplay
9640              {\glentrysymbol{\glslabel}}{\glslabel}}%
9641          {\glsinsert}}%
9642      }%
9643      }%
9644      }%

```

```

9645 }%
9646 {%

    Custom text provided in \glsdisp
9647     \ifglsused{\glslabel}%
9648     {%

        Subsequent use
9649         #2{\glscustomtext}%
9650         {\glsdescriptionaccessdisplay
9651          {\glsentrydesc{\glslabel}}{\glslabel}}%
9652         {\glssymbolaccessdisplay
9653          {\glsentrysymbol{\glslabel}}{\glslabel}}%
9654         {\glsinsert}%
9655     }%
9656     {%

        First use
9657         #1{\glscustomtext}%
9658         {\glsdescriptionaccessdisplay
9659          {\glsentrydesc{\glslabel}}{\glslabel}}%
9660         {\glssymbolaccessdisplay
9661          {\glsentrysymbol{\glslabel}}{\glslabel}}%
9662         {\glsinsert}%
9663     }%
9664 }%
9665 }

```

`\glsgenentryfmt` Redefine to use accessibility information.

```

9666 \renewcommand*{\glsgenentryfmt}{%
9667     \ifdefempty\glscustomtext
9668     {%
9669         \glsifplural
9670         {%

            Plural form
9671             \glscapscase
9672             {%

                Don't adjust case
9673                 \ifglsused\glslabel
9674                 {%

                    Subsequent use
9675                     \glspluralaccessdisplay
9676                     {\glsentryplural{\glslabel}}{\glslabel}%
9677                     \glsinsert
9678                 }%
9679                 {%

                    First use
9680                     \glsfirstpluralaccessdisplay

```

```

9681          {\glsentryfirstplural{\glslabel}}{\glslabel}%
9682      \glsinsert
9683  }%
9684  }%
9685  {%

```

Make first letter upper case

```

9686      \ifglsused\glslabel
9687  {%

```

Subsequent use.

```

9688      \glspluralaccessdisplay
9689      {\Glsentryplural{\glslabel}}{\glslabel}%
9690      \glsinsert
9691  }%
9692  {%

```

First use

```

9693      \glsfirstpluralaccessdisplay
9694      {\Glsentryfirstplural{\glslabel}}{\glslabel}%
9695      \glsinsert
9696  }%
9697  }%
9698  {%

```

Make all upper case

```

9699      \ifglsused\glslabel
9700  {%

```

Subsequent use

```

9701      \glspluralaccessdisplay
9702      {\mfirstucMakeUppercase{\glsentryplural{\glslabel}}}%
9703      {\glslabel}%
9704      \mfirstucMakeUppercase{\glsinsert}%
9705  }%
9706  {%

```

First use

```

9707      \glsfirstpluralaccessdisplay
9708      {\mfirstucMakeUppercase{\glsentryfirstplural{\glslabel}}}%
9709      {\glslabel}%
9710      \mfirstucMakeUppercase{\glsinsert}%
9711  }%
9712  }%
9713  }%
9714  {%

```

Singular form

```

9715      \glscapscase
9716  {%

```

Don't adjust case



```

9717      \ifglsused\glslabel
9718      {%

```

Subsequent use

```

9719      \glstextaccessdisplay{\glsentrytext{\glslabel}}{\glslabel}%
9720      \glsinsert
9721      }%
9722      {%

```

First use

```

9723      \glsfirstaccessdisplay{\glsentryfirst{\glslabel}}{\glslabel}%
9724      \glsinsert
9725      }%
9726      }%
9727      {%

```

Make first letter upper case

```

9728      \ifglsused\glslabel
9729      {%

```

Subsequent use

```

9730      \glstextaccessdisplay{\Glsentrytext{\glslabel}}{\glslabel}%
9731      \glsinsert
9732      }%
9733      {%

```

First use

```

9734      \glsfirstaccessdisplay{\Glsentryfirst{\glslabel}}{\glslabel}%
9735      \glsinsert
9736      }%
9737      }%
9738      {%

```

Make all upper case

```

9739      \ifglsused\glslabel
9740      {%

```

Subsequent use

```

9741      \glstextaccessdisplay
9742      {\mfirstucMakeUppercase{\glsentrytext{\glslabel}}}{\glslabel}%
9743      \mfirstucMakeUppercase{\glsinsert}%
9744      }%
9745      {%

```

First use

```

9746      \glsfirstaccessdisplay
9747      {\mfirstucMakeUppercase{\glsentryfirst{\glslabel}}}{\glslabel}%
9748      \mfirstucMakeUppercase{\glsinsert}%
9749      }%
9750      }%
9751      }%
9752      }%
9753      {%

```

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.

```
9754 \glscustomtext\glsinsert
9755 }%
9756 }
```

`\glsngenacfmt` Redefine to include accessibility information.

```
9757 \renewcommand*{\glsngenacfmt}{%
9758 \ifdefempty\glscustomtext
9759 {%
9760 \ifglssused\glslabel
9761 {%
```

Subsequent use:

```
9762 \glsifplural
9763 {%
```

Subsequent plural form:

```
9764 \glscapscase
9765 {%
```

Subsequent plural form, don't adjust case:

```
9766 \acronymfont
9767 {\glsshortpluralaccessdisplay
9768 {\glssentryshortpl{\glslabel}}{\glslabel}}%
9769 \glsinsert
9770 }%
9771 {%
```

Subsequent plural form, make first letter upper case:

```
9772 \acronymfont
9773 {\glsshortpluralaccessdisplay
9774 {\Glsentryshortpl{\glslabel}}{\glslabel}}%
9775 \glsinsert
9776 }%
9777 {%
```

Subsequent plural form, all caps:

```
9778 \mfirstucMakeUppercase
9779 {\acronymfont
9780 {\glsshortpluralaccessdisplay
9781 {\glssentryshortpl{\glslabel}}{\glslabel}}%
9782 \glsinsert}%
9783 }%
9784 }%
9785 {%
```

Subsequent singular form

```
9786 \glscapscase
9787 {%
```

Subsequent singular form, don't adjust case:

```
9788      \acronymfont
9789      {\glsshortaccessdisplay{\glsentryshort{\glslabel}}{\glslabel}}%
9790      \glsinsert
9791      }%
9792      {%
```

Subsequent singular form, make first letter upper case:

```
9793      \acronymfont
9794      {\glsshortaccessdisplay{\Glsentryshort{\glslabel}}{\glslabel}}%
9795      \glsinsert
9796      }%
9797      {%
```

Subsequent singular form, all caps:

```
9798      \mfirstucMakeUppercase
9799      {\acronymfont{%
9800      \glsshortaccessdisplay{\glsentryshort{\glslabel}}{\glslabel}}%
9801      \glsinsert}%
9802      }%
9803      }%
9804      }%
9805      {%
```

First use:

```
9806      \glsifplural
9807      {%
```

First use plural form:

```
9808      \glscapscase
9809      {%
```

First use plural form, don't adjust case:

```
9810      \genplacrfullformat{\glslabel}{\glsinsert}%
9811      }%
9812      {%
```

First use plural form, make first letter upper case:

```
9813      \Genplacrfullformat{\glslabel}{\glsinsert}%
9814      }%
9815      {%
```

First use plural form, all caps:

```
9816      \mfirstucMakeUppercase
9817      {\genplacrfullformat{\glslabel}{\glsinsert}}%
9818      }%
9819      }%
9820      {%
```

First use singular form

```
9821      \glscapscase
9822      {%
```

First use singular form, don't adjust case:

```
9823      \genacrfullformat{\glslabel}{\glsinsert}%
9824      }%
9825      {%
```

First use singular form, make first letter upper case:

```
9826      \Genacrfullformat{\glslabel}{\glsinsert}%
9827      }%
9828      {%
```

First use singular form, all caps:

```
9829      \mfirstucMakeUppercase
9830      {\genacrfullformat{\glslabel}{\glsinsert}}%
9831      }%
9832      }%
9833      }%
9834      }%
9835      {%
```

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.

```
9836      \glscustomtext
9837      }%
9838 }
```

`\genacrfullformat` Redefine to include accessibility information.

```
9839 \renewcommand*{\genacrfullformat}[2]{%
9840   \glslongaccessdisplay{\glsentrylong{#1}}{#1}#2\space
9841   (\glsshortaccessdisplay{\protect\firstacronymfont{\glsentryshort{#1}}}{#1})%
9842 }
```

`\Genacrfullformat` Redefine to include accessibility information.

```
9843 \renewcommand*{\Genacrfullformat}[2]{%
9844   \glslongaccessdisplay{\Glsentrylong{#1}}{#1}#2\space
9845   (\glsshortaccessdisplay{\protect\firstacronymfont{\Glsentryshort{#1}}}{#1})%
9846 }
```

`\genplacrfullformat` Redefine to include accessibility information.

```
9847 \renewcommand*{\genplacrfullformat}[2]{%
9848   \glslongpluralaccessdisplay{\glsentrylongpl{#1}}{#1}#2\space
9849   (\glsshortpluralaccessdisplay
9850     {\protect\firstacronymfont{\glsentryshortpl{#1}}}{#1})%
9851 }
```

`\Genplacrfullformat` Redefine to include accessibility information.

```
9852 \renewcommand*{\Genplacrfullformat}[2]{%
9853   \glslongpluralaccessdisplay{\Glsentrylongpl{#1}}{#1}#2\space
9854   (\glsshortpluralaccessdisplay
9855     {\protect\firstacronymfont{\Glsentryshortpl{#1}}}{#1})%
9856 }
```

\@acrshort

```

9857 \def\@acrshort#1#2[#3]{%
9858   \glsdoifexists{#2}%
9859   {%
9860     \let\do@gl@link@checkfirsthyper\relax

9861     \let\glsifplural\@secondoftwo
9862     \let\glscapscase\@firstofthree
9863     \let\glsinsert\@empty
9864     \def\glscustomtext{%
9865       \acronymfont{\glsshortaccessdisplay{\glsentryshort{#2}}{#2}}#3%
9866     }%

    Call \@gl@link

9867     \@gl@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
9868   }%
9869 }
```

\@Acrshort

```

9870 \def\@Acrshort#1#2[#3]{%
9871   \glsdoifexists{#2}%
9872   {%
9873     \let\do@gl@link@checkfirsthyper\relax

9874     \let\glsifplural\@secondoftwo
9875     \let\glscapscase\@secondofthree
9876     \let\glsinsert\@empty
9877     \def\glscustomtext{%
9878       \acronymfont{\glsshortaccessdisplay{\Glsentryshort{#2}}{#2}}#3%
9879     }%

    Call \@gl@link

9880     \@gl@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
9881   }%
9882 }
```

\@ACRshort

```

9883 \def\@ACRshort#1#2[#3]{%
9884   \glsdoifexists{#2}%
9885   {%
9886     \let\do@gl@link@checkfirsthyper\relax

9887     \let\glsifplural\@secondoftwo
9888     \let\glscapscase\@thirdofthree
9889     \let\glsinsert\@empty
9890     \def\glscustomtext{%
9891       \acronymfont{\glsshortaccessdisplay
9892         {\MakeUppercase{\glsentryshort{#2}}}{#2}}#3%
9893     }%
9894 }
```

```

Call \@gls@link
9894   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
9895   }%
9896 }

```

\@acrlong

```

9897 \def\@acrlong#1#2[#3]{%
9898   \glsdoifexists{#2}%
9899   {%
9900     \let\do@gls@link@checkfirsthyper\relax

9901     \let\glsifplural\@secondoftwo
9902     \let\glscapscase\@firstofthree
9903     \let\glsinsert\@empty
9904     \def\glscustomtext{%
9905       \acronymfont{\glslongaccessdisplay{\glsentrylong{#2}}{#2}}#3%
9906     }%

```

```

Call \@gls@link
9907   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
9908   }%
9909 }

```

\@Acrlong

```

9910 \def\@Acrlong#1#2[#3]{%
9911   \glsdoifexists{#2}%
9912   {%
9913     \let\do@gls@link@checkfirsthyper\relax

9914     \let\glsifplural\@secondoftwo
9915     \let\glscapscase\@firstofthree
9916     \let\glsinsert\@empty
9917     \def\glscustomtext{%
9918       \acronymfont{\glslongaccessdisplay{\Glsentrylong{#2}}{#2}}#3%
9919     }%

```

```

Call \@gls@link
9920   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
9921   }%
9922 }

```

\@ACRlong

```

9923 \def\@ACRlong#1#2[#3]{%
9924   \glsdoifexists{#2}%
9925   {%
9926     \let\do@gls@link@checkfirsthyper\relax

9927     \let\glsifplural\@secondoftwo
9928     \let\glscapscase\@firstofthree
9929     \let\glsinsert\@empty

```

```

9930 \def\glscustomtext{%
9931 \acronymfont{\glslongaccessdisplay{%
9932 \MakeUppercase{\glsentrylong{#2}}}{#2}#3}%
9933 }%

Call \@gls@link
9934 \@gls@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
9935 }%
9936 }

```

### 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.

```

9937 \renewcommand*{\glossentryname}[1]{%
9938 \glsdoifexists{#1}%
9939 {%
9940 \glsnamefont{\glsnameaccessdisplay{\glsentryname{#1}}{#1}}%
9941 }%
9942 }

9943 \renewcommand*{\glossentryname}[1]{%
9944 \glsdoifexists{#1}%
9945 {%
9946 \glsnamefont{\glsnameaccessdisplay{\Glsentryname{#1}}{#1}}%
9947 }%
9948 }

9949 \renewcommand*{\glossentrydesc}[1]{%
9950 \glsdoifexists{#1}%
9951 {%
9952 \glsdescriptionaccessdisplay{\glsentrydesc{#1}}{#1}%
9953 }%
9954 }

9955 \renewcommand*{\Glossentrydesc}[1]{%
9956 \glsdoifexists{#1}%
9957 {%
9958 \glsdescriptionaccessdisplay{\Glsentrydesc{#1}}{#1}%
9959 }%
9960 }

9961 \renewcommand*{\glossentrysymbol}[1]{%
9962 \glsdoifexists{#1}%
9963 {%

```

```

9964 \glssymbolaccessdisplay{\glentrysymbol{#1}}{#1}%
9965 }%
9966 }

9967 \renewcommand*{\Glossentrysymbol}[1]{%
9968 \glsoifexists{#1}%
9969 {%
9970 \glssymbolaccessdisplay{\Glentrysymbol{#1}}{#1}%
9971 }%
9972 }

```

#### pglossaryentryfield

```

9973 \newcommand*{\accsuppglossaryentryfield}[5]{%
9974 \glossaryentryfield{#1}%
9975 {\glnameaccessdisplay{#2}{#1}}%
9976 {\gldescriptionaccessdisplay{#3}{#1}}%
9977 {\glssymbolaccessdisplay{#4}{#1}}{#5}%
9978 }

```

#### glossarysubentryfield

```

9979 \newcommand*{\accsuppglossarysubentryfield}[6]{%
9980 \glossarysubentryfield{#1}{#2}%
9981 {\glnameaccessdisplay{#3}{#2}}%
9982 {\gldescriptionaccessdisplay{#4}{#2}}%
9983 {\glssymbolaccessdisplay{#5}{#2}}{#6}%
9984 }

```

## 7.4 Acronyms

Redefine acronym styles provided by glossaries:

long-short *<long>* (*<short>*) acronym style.

```

9985 \renewacronymstyle{long-short}%
9986 {%

```

Check for long form in case this is a mixed glossary.

```

9987 \ifglshaslong{\glslabel}{\glsgenacfmt}{\glsgenentryfmt}%
9988 }%
9989 {%
9990 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
9991 \renewcommand*{\genacrfullformat}[2]{%
9992 \glslongaccessdisplay{\glentrylong{##1}}{##1}##2\space
9993 (\glsshortaccessdisplay
9994 {\protect\firstacronymfont{\glentryshort{##1}}}{##1})%
9995 }%
9996 \renewcommand*{\Genacrfullformat}[2]{%
9997 \glslongaccessdisplay{\Glentrylong{##1}}{##1}##2\space
9998 (\glsshortaccessdisplay
9999 {\protect\firstacronymfont{\glentryshort{##1}}}{##1})%
10000 }%

```



```

10001 \renewcommand*{\genplacrfullformat}[2]{%
10002 \glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1}##2\space
10003 (\glsshortpluralaccessdisplay
10004 {\protect\firstacronymfont{\glsentryshortpl{##1}}{##1}})%
10005 }%
10006 \renewcommand*{\Genplacrfullformat}[2]{%
10007 \glslongpluralaccessdisplay{\Glsentrylongpl{##1}}{##1}##2\space
10008 (\glsshortpluralaccessdisplay
10009 {\protect\firstacronymfont{\glsentryshortpl{##1}}{##1}})%
10010 }%
10011 \renewcommand*{\acronymentry}[1]{%
10012 \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}{##1}}
10013 \renewcommand*{\acronymsort}[2]{##1}%
10014 \renewcommand*{\acronymfont}[1]{##1}%
10015 \renewcommand*{\firstacronymfont}[1]{\acronymfont{##1}}%
10016 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
10017 }

```

short-long <short> (<long>) acronym style.

```

10018 \renewacronymstyle{short-long}%
10019 {%
10020 \ifglshaslong{\glslabel}{\glsгенacfmt}{\glsгенentryfmt}%
10021 }%
10022 {%
10023 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
10024 \renewcommand*{\genacrfullformat}[2]{%
10025 \glsshortaccessdisplay
10026 {\protect\firstacronymfont{\glsentryshort{##1}}{##1}##2\space
10027 (\glslongaccessdisplay{\glsentrylong{##1}}{##1}})%
10028 }%
10029 \renewcommand*{\Genacrfullformat}[2]{%
10030 \glsshortaccessdisplay
10031 {\protect\firstacronymfont{\Glsentryshort{##1}}{##1}##2\space
10032 (\glslongaccessdisplay{\glsentrylong{##1}}{##1}})%
10033 }%
10034 \renewcommand*{\genplacrfullformat}[2]{%
10035 \glsshortpluralaccessdisplay
10036 {\protect\firstacronymfont{\glsentryshortpl{##1}}{##1}##2\space
10037 (\glslongpluralaccessdisplay
10038 {\glsentrylongpl{##1}}{##1}})%
10039 }%
10040 \renewcommand*{\Genplacrfullformat}[2]{%
10041 \glsshortpluralaccessdisplay
10042 {\protect\firstacronymfont{\Glsentryshortpl{##1}}{##1}##2\space
10043 (\glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1}})%
10044 }%
10045 \renewcommand*{\acronymentry}[1]{%
10046 \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}{##1}}%

```

```

10047 \renewcommand*{\acronymsort}[2]{##1}%
10048 \renewcommand*{\acronymfont}[1]{##1}%
10049 \renewcommand*{\firstacronymfont}[1]{\acronymfont{##1}}%
10050 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
10051 }

```

long-short-desc    *⟨long⟩* (*⟨short⟩*) acronym style that has an accompanying description (which the user needs to supply).

```

10052 \renewacronymstyle{long-short-desc}%
10053 {%
10054   \GlsUseAcrEntryDisplayStyle{long-short}%
10055 }%
10056 {%
10057   \GlsUseAcrStyleDefs{long-short}%
10058   \renewcommand*{\GenericAcronymFields}{}%
10059   \renewcommand*{\acronymsort}[2]{##2}%
10060   \renewcommand*{\acronymentry}[1]{%
10061     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10062     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10063 }

```

long-sc-short-desc    *⟨long⟩* (*\textsc{⟨short⟩}*) acronym style that has an accompanying description (which the user needs to supply).

```

10064 \renewacronymstyle{long-sc-short-desc}%
10065 {%
10066   \GlsUseAcrEntryDisplayStyle{long-sc-short}%
10067 }%
10068 {%
10069   \GlsUseAcrStyleDefs{long-sc-short}%
10070   \renewcommand*{\GenericAcronymFields}{}%
10071   \renewcommand*{\acronymsort}[2]{##2}%
10072   \renewcommand*{\acronymentry}[1]{%
10073     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10074     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10075 }

```

long-sm-short-desc    *⟨long⟩* (*\textsmaller{⟨short⟩}*) acronym style that has an accompanying description (which the user needs to supply).

```

10076 \renewacronymstyle{long-sm-short-desc}%
10077 {%
10078   \GlsUseAcrEntryDisplayStyle{long-sm-short}%
10079 }%
10080 {%
10081   \GlsUseAcrStyleDefs{long-sm-short}%
10082   \renewcommand*{\GenericAcronymFields}{}%
10083   \renewcommand*{\acronymsort}[2]{##2}%
10084   \renewcommand*{\acronymentry}[1]{%
10085     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10086     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%

```

10087 }

short-long-desc    *<short>* (*{<long>}*) acronym style that has an accompanying description (which the user needs to supply).

```
10088 \renewacronymstyle{short-long-desc}%
10089 {%
10090   \GlsUseAcrEntryDispStyle{short-long}%
10091 }%
10092 {%
10093   \GlsUseAcrStyleDefs{short-long}%
10094   \renewcommand*{\GenericAcronymFields}{}%
10095   \renewcommand*{\acronymsort}[2]{##2}%
10096   \renewcommand*{\acronymentry}[1]{%
10097     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10098     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10099 }
```

sc-short-long-desc    *<long>* (*\textsc{<short>}*) acronym style that has an accompanying description (which the user needs to supply).

```
10100 \renewacronymstyle{sc-short-long-desc}%
10101 {%
10102   \GlsUseAcrEntryDispStyle{sc-short-long}%
10103 }%
10104 {%
10105   \GlsUseAcrStyleDefs{sc-short-long}%
10106   \renewcommand*{\GenericAcronymFields}{}%
10107   \renewcommand*{\acronymsort}[2]{##2}%
10108   \renewcommand*{\acronymentry}[1]{%
10109     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10110     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10111 }
```

sm-short-long-desc    *<long>* (*\textsmaller{<short>}*) acronym style that has an accompanying description (which the user needs to supply).

```
10112 \renewacronymstyle{sm-short-long-desc}%
10113 {%
10114   \GlsUseAcrEntryDispStyle{sm-short-long}%
10115 }%
10116 {%
10117   \GlsUseAcrStyleDefs{sm-short-long}%
10118   \renewcommand*{\GenericAcronymFields}{}%
10119   \renewcommand*{\acronymsort}[2]{##2}%
10120   \renewcommand*{\acronymentry}[1]{%
10121     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10122     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10123 }
```

dua    *<long>* only acronym style.

10124 \renewacronymstyle{dua}%

10125 {%

Check for long form in case this is a mixed glossary.

10126 \ifdefempty\glscustomtext

10127 {%

10128 \ifglshaslong{\glslabel}%

10129 {%

10130 \glsifplural

10131 {%

Plural form:

10132 \glscapscase

10133 {%

Plural form, don't adjust case:

10134 \glslongpluralaccessdisplay{\glentrylongpl{\glslabel}}{\glslabel}%

10135 \glsinsert

10136 }%

10137 {%

Plural form, make first letter upper case:

10138 \glslongpluralaccessdisplay{\Glsentrylongpl{\glslabel}}{\glslabel}%

10139 \glsinsert

10140 }%

10141 {%

Plural form, all caps:

10142 \glslongpluralaccessdisplay

10143 {\mfirstucMakeUppercase{\glentrylongpl{\glslabel}}}{\glslabel}%

10144 \mfirstucMakeUppercase{\glsinsert}%

10145 }%

10146 }%

10147 {%

Singular form

10148 \glscapscase

10149 {%

Singular form, don't adjust case:

10150 \glslongaccessdisplay{\glentrylong{\glslabel}}{\glslabel}\glsinsert

10151 }%

10152 {%

Subsequent singular form, make first letter upper case:

10153 \glslongaccessdisplay{\Glsentrylong{\glslabel}}{\glslabel}\glsinsert

10154 }%

10155 {%

Subsequent singular form, all caps:

10156 \glslongaccessdisplay

10157 {\mfirstucMakeUppercase

10158 {\glentrylong{\glslabel}\glsinsert}}{\glslabel}%

```

10159         \mfirstucMakeUppercase{\glsinsert}%
10160     }%
10161 }%
10162 }%
10163 {%

    Not an acronym:
10164     \glsgenentryfmt
10165 }%
10166 }%
10167 {\glscustomtext\glsinsert}%
10168 }%
10169 {%
10170 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
10171 \renewcommand*{\acrfullfmt}[3]{%
10172     \glslink[##1]{##2}{%
10173         \glslongaccessdisplay{\glsentrylong{##2}}{##2}##3\space
10174         (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##2}}}{##2}}}%
10175 \renewcommand*{\Acrfullfmt}[3]{%
10176     \glslink[##1]{##2}{%
10177         \glslongaccessdisplay{\Glsentrylong{##2}}{##2}##3\space
10178         (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##2}}}{##2}}}%
10179 \renewcommand*{\ACRfullfmt}[3]{%
10180     \glslink[##1]{##2}{%
10181         \glslongaccessdisplay
10182         {\mfirstucMakeUppercase{\glsentrylong{##2}}{##2}##3\space
10183         (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##2}}}{##2}})}}%
10184 \renewcommand*{\acrfullplfmt}[3]{%
10185     \glslink[##1]{##2}{%
10186         \glslongpluralaccessdisplay
10187         {\glsentrylongpl{##2}}{##2}##3\space
10188         (\glsshortpluralaccessdisplay
10189         {\acronymfont{\glsentryshortpl{##2}}}{##2}})}}%
10190 \renewcommand*{\Acrfullplfmt}[3]{%
10191     \glslink[##1]{##2}{%
10192         \glslongpluralaccessdisplay
10193         {\Glsentrylongpl{##2}}{##2}##3\space
10194         (\glsshortpluralaccessdisplay
10195         {\acronymfont{\glsentryshortpl{##2}}}{##2}})}}%
10196 \renewcommand*{\ACRfullplfmt}[3]{%
10197     \glslink[##1]{##2}{%
10198         \glslongpluralaccessdisplay
10199         {\mfirstucMakeUppercase{\glsentrylongpl{##2}}{##2}##3\space
10200         (\glsshortpluralaccessdisplay
10201         {\acronymfont{\glsentryshortpl{##2}}}{##2}})}}%
10202 \renewcommand*{\glsentryfull}[1]{%
10203     \glslongaccessdisplay{\glsentrylong{##1}}\space
10204     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})%
10205 }%
10206 \renewcommand*{\Glsentryfull}[1]{%

```

```

10207 \glslongaccessdisplay{\Glsentrylong{##1}}{##1}\space
10208 (\glsshortaccessdisplay{\acronymfont{\glentryshort{##1}}}{##1})%
10209 }%
10210 \renewcommand*{\glsentryfullpl}[1]{%
10211 \glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1}\space
10212 (\glsshortpluralaccessdisplay{\acronymfont{\glentryshortpl{##1}}}{##1})%
10213 }%
10214 \renewcommand*{\Glsentryfullpl}[1]{%
10215 \glslongpluralaccessdisplay{\Glsentrylongpl{##1}}{##1}\space
10216 (\glsshortpluralaccessdisplay{\acronymfont{\glentryshortpl{##1}}}{##1})%
10217 }%
10218 \renewcommand*{\acronymentry}[1]{%
10219 \glsshortaccessdisplay{\acronymfont{\glentryshort{##1}}}{##1}%
10220 \renewcommand*{\acronymsort}[2]{##1}%
10221 \renewcommand*{\acronymfont}[1]{##1}%
10222 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
10223 }

```

dua-desc *<long>* only acronym style with user-supplied description.

```

10224 \renewacronymstyle{dua-desc}%
10225 {%
10226 \GlsUseAcrEntryDispStyle{dua}%
10227 }%
10228 {%
10229 \GlsUseAcrStyleDefs{dua}%
10230 \renewcommand*{\GenericAcronymFields}{}%
10231 \renewcommand*{\acronymentry}[1]{%
10232 \glslongaccessdisplay{\acronymfont{\glentrylong{##1}}}{##1}%
10233 \renewcommand*{\acronymsort}[2]{##2}%
10234 }%

```

footnote *<short>*\footnote{*<long>*} acronym style.

```

10235 \renewacronymstyle{footnote}%
10236 {%

```

Check for long form in case this is a mixed glossary.

```

10237 \ifglshaslong{\glslabel}{\glsgenacfmt}{\glsgenentryfmt}%
10238 }%
10239 {%
10240 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%

```

Need to ensure hyperlinks are switched off on first use:

```

10241 \glshyperfirstfalse
10242 \renewcommand*{\genacrfullformat}[2]{%
10243 \glsshortaccessdisplay
10244 {\protect\firstacronymfont{\glentryshort{##1}}}{##1}##2%
10245 \protect\footnote{\glslongaccessdisplay{\glentrylong{##1}}{##1}}%
10246 }%
10247 \renewcommand*{\Genacrfullformat}[2]{%
10248 \glsshortaccessdisplay

```

```

10249     {\firstacronymfont{\Glsentryshort{##1}}}{##1}##2%
10250     \protect\footnote{\glslongaccessdisplay{\glsentrylong{##1}}{##1}}%
10251 }%
10252 \renewcommand*{\genplacrfullformat}[2]{%
10253     \glsshortpluralaccessdisplay
10254     {\protect\firstacronymfont{\glsentryshortpl{##1}}}{##1}##2%
10255     \protect\footnote{\glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1}}%
10256 }%
10257 \renewcommand*{\Genplacrfullformat}[2]{%
10258     \glsshortpluralaccessdisplay
10259     {\protect\firstacronymfont{\Glsentryshortpl{##1}}}{##1}##2%
10260     \protect\footnote{\glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1}}%
10261 }%
10262 \renewcommand*{\acronymentry}[1]{%
10263     \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}}%
10264 \renewcommand*{\acronymsort}[2]{##1}%
10265 \renewcommand*{\acronymfont}[1]{##1}%
10266 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%

```

Don't use footnotes for \acrfull:

```

10267 \renewcommand*{\acrfullfmt}[3]{%
10268     \glslink[##1]{##2}{%
10269         \glsshortaccessdisplay{\acronymfont{\glsentryshort{##2}}}{##2}##3\space
10270         (\glslongaccessdisplay{\glsentrylong{##2}}{##2})}%
10271 \renewcommand*{\Acrfullfmt}[3]{%
10272     \glslink[##1]{##2}{%
10273         \glsshortaccessdisplay{\acronymfont{\Glsentryshort{##2}}}{##2}##3\space
10274         (\glslongaccessdisplay{\glsentrylong{##2}}{##2})}%
10275 \renewcommand*{\ACRfullfmt}[3]{%
10276     \glslink[##1]{##2}{%
10277         \glsshortaccessdisplay
10278         {\mfirstucMakeUppercase
10279         {\acronymfont{\glsentryshort{##2}}}{##2}##3\space
10280         (\glslongaccessdisplay{\glsentrylong{##2}}{##2})}%
10281 \renewcommand*{\acrfullplfmt}[3]{%
10282     \glslink[##1]{##2}{%
10283         \glsshortpluralaccessdisplay
10284         {\acronymfont{\glsentryshortpl{##2}}}{##2}##3\space
10285         (\glslongpluralaccessdisplay{\glsentrylongpl{##2}}{##2})}%
10286 \renewcommand*{\Acrfullplfmt}[3]{%
10287     \glslink[##1]{##2}{%
10288         \glsshortpluralaccessdisplay
10289         {\acronymfont{\Glsentryshortpl{##2}}}{##2}##3\space
10290         (\glslongpluralaccessdisplay{\glsentrylongpl{##2}}{##2})}%
10291 \renewcommand*{\ACRfullplfmt}[3]{%
10292     \glslink[##1]{##2}{%
10293         \glsshortpluralaccessdisplay
10294         {\mfirstucMakeUppercase
10295         {\acronymfont{\glsentryshortpl{##2}}}{##2}##3\space
10296         (\glslongpluralaccessdisplay{\glsentrylongpl{##2}}{##2})}%

```

Similarly for \glsentryfull etc:

```

10297 \renewcommand*{\glsentryfull}[1]{%
10298   \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}\space
10299   (\glslongaccessdisplay{\glsentrylong{##1}}{##1})}%
10300 \renewcommand*{\Glsentryfull}[1]{%
10301   \glsshortaccessdisplay{\acronymfont{\Glsentryshort{##1}}}{##1}\space
10302   (\glslongaccessdisplay{\glsentrylong{##1}}{##1})}%
10303 \renewcommand*{\glsentryfullpl}[1]{%
10304   \glsshortpluralaccessdisplay
10305   {\acronymfont{\glsentryshortpl{##1}}}{##1}\space
10306   (\glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1})}%
10307 \renewcommand*{\Glsentryfullpl}[1]{%
10308   \glsshortpluralaccessdisplay
10309   {\acronymfont{\Glsentryshortpl{##1}}}{##1}\space
10310   (\glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1})}%
10311 }

```

footnote-sc \textsc{<short>}\footnote{<long>} acronym style.

```

10312 \renewacronymstyle{footnote-sc}%
10313 {%
10314   \GlsUseAcrEntryDispStyle{footnote}%
10315 }%
10316 {%
10317   \GlsUseAcrStyleDefs{footnote}%
10318   \renewcommand{\acronymentry}[1]{%
10319     \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}}
10320   \renewcommand{\acronymfont}[1]{\textsc{##1}}%
10321   \renewcommand*{\acrpluralsuffix}{\glstextup{\glspluralsuffix}}%
10322 }%

```

footnote-sm \textsmaller{<short>}\footnote{<long>} acronym style.

```

10323 \renewacronymstyle{footnote-sm}%
10324 {%
10325   \GlsUseAcrEntryDispStyle{footnote}%
10326 }%
10327 {%
10328   \GlsUseAcrStyleDefs{footnote}%
10329   \renewcommand{\acronymentry}[1]{%
10330     \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}}
10331   \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
10332   \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
10333 }%

```

footnote-desc <short>\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```

10334 \renewacronymstyle{footnote-desc}%
10335 {%
10336   \GlsUseAcrEntryDispStyle{footnote}%
10337 }%

```



```

10338 {%
10339   \GlsUseAcrStyleDefs{footnote}%
10340   \renewcommand*{\GenericAcronymFields}{}%
10341   \renewcommand*{\acronymsort}[2]{##2}%
10342   \renewcommand*{\acronymentry}[1]{%
10343     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10344     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10345 }

```

footnote-sc-desc \textsc{<short>}\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```

10346 \renewacronymstyle{footnote-sc-desc}%
10347 {%
10348   \GlsUseAcrEntryDispStyle{footnote-sc}%
10349 }%
10350 {%
10351   \GlsUseAcrStyleDefs{footnote-sc}%
10352   \renewcommand*{\GenericAcronymFields}{}%
10353   \renewcommand*{\acronymsort}[2]{##2}%
10354   \renewcommand*{\acronymentry}[1]{%
10355     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10356     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10357 }

```

footnote-sm-desc \textsmaller{<short>}\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```

10358 \renewacronymstyle{footnote-sm-desc}%
10359 {%
10360   \GlsUseAcrEntryDispStyle{footnote-sm}%
10361 }%
10362 {%
10363   \GlsUseAcrStyleDefs{footnote-sm}%
10364   \renewcommand*{\GenericAcronymFields}{}%
10365   \renewcommand*{\acronymsort}[2]{##2}%
10366   \renewcommand*{\acronymentry}[1]{%
10367     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10368     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10369 }

```

Use \newacronymhook to modify the key list to set the access text to the long version by default.

```

10370 \renewcommand*{\newacronymhook}{%
10371   \edef\@gls@keylist{shortaccess=\the\glslongtok,%
10372     \the\glskeylisttok}%
10373   \expandafter\glskeylisttok\expandafter{\@gls@keylist}%
10374 }

```

defaultNewAcronymDef Modify default style to use access text:

```

10375 \renewcommand*{\DefaultNewAcronymDef}{%
10376   \edef\@do@newglossaryentry{%
10377     \noexpand\newglossaryentry{\the\glslabeltok}%
10378     {%
10379       type=\acronymtype,%
10380       name={\the\glsshorttok},%
10381       description={\the\glslongtok},%
10382       descriptionaccess=\relax,
10383       text={\the\glsshorttok},%
10384       access={\noexpand\@glo@textaccess},%
10385       sort={\the\glsshorttok},%
10386       short={\the\glsshorttok},%
10387       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10388       shortaccess={\the\glslongtok},%
10389       long={\the\glslongtok},%
10390       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10391       descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10392       first={\noexpand\glslongaccessdisplay
10393         {\the\glslongtok}{\the\glslabeltok}\space
10394         (\noexpand\glsshortaccessdisplay
10395           {\the\glsshorttok}{\the\glslabeltok})},%
10396       plural={\the\glsshorttok\acrpluralsuffix},%
10397       firstplural={\noexpand\glslongpluralaccessdisplay
10398         {\noexpand\@glo@longpl}{\the\glslabeltok}\space
10399         (\noexpand\glsshortpluralaccessdisplay
10400           {\noexpand\@glo@shortpl}{\the\glslabeltok})},%
10401       firstaccess=\relax,
10402       firstpluralaccess=\relax,
10403       textaccess={\noexpand\@glo@shortaccess},%
10404       \the\glskeylisttok
10405     }%
10406   }%
10407   \let\@org@gls@assign@firstpl\gls@assign@firstpl
10408   \let\@org@gls@assign@plural\gls@assign@plural
10409   \let\@org@gls@assign@descplural\gls@assign@descplural
10410   \def\gls@assign@firstpl##1##2{%
10411     \@@gls@expand@field{##1}{firstpl}{##2}%
10412   }%
10413   \def\gls@assign@plural##1##2{%
10414     \@@gls@expand@field{##1}{plural}{##2}%
10415   }%
10416   \def\gls@assign@descplural##1##2{%
10417     \@@gls@expand@field{##1}{descplural}{##2}%
10418   }%
10419   \@do@newglossaryentry
10420   \let\gls@assign@firstpl\@org@gls@assign@firstpl
10421   \let\gls@assign@plural\@org@gls@assign@plural
10422   \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
10423 }

```

otnoteNewAcronymDef

```

10424 \renewcommand*{\DescriptionFootnoteNewAcronymDef}{%
10425   \edef\@do@newglossaryentry{%
10426     \noexpand\newglossaryentry{\the\glslabeltok}%
10427     {%
10428       type=\acronymtype,%
10429       name={\noexpand\acronymfont{\the\glsshorttok}},%
10430       sort={\the\glsshorttok},%
10431       text={\the\glsshorttok},%
10432       short={\the\glsshorttok},%
10433       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10434       shortaccess={\the\glslongtok},%
10435       long={\the\glslongtok},%
10436       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10437       access={\noexpand\@glo@textaccess},%
10438       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10439       symbol={\the\glslongtok},%
10440       symbolplural={\the\glslongtok\noexpand\acrpluralsuffix},%
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@symbolplural\gls@assign@symbolplural
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@symbolplural##1##2{%
10456     \@@gls@expand@field{##1}{symbolplural}{##2}%
10457   }%
10458   \@do@newglossaryentry
10459   \let\gls@assign@plural\@org@gls@assign@plural
10460   \let\gls@assign@firstpl\@org@gls@assign@firstpl
10461   \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
10462 }

```

ipitionNewAcronymDef

```

10463 \renewcommand*{\DescriptionNewAcronymDef}{%
10464   \edef\@do@newglossaryentry{%
10465     \noexpand\newglossaryentry{\the\glslabeltok}%
10466     {%
10467       type=\acronymtype,%
10468       name={\noexpand
10469         \acrnameformat{\the\glsshorttok}{\the\glslongtok}},%

```

```

10470     access={\noexpand\@glo@textaccess},%
10471     sort={\the\glsshorttok},%
10472     short={\the\glsshorttok},%
10473     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10474     shortaccess={\the\glslongtok},%
10475     long={\the\glslongtok},%
10476     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10477     first={\the\glslongtok},%
10478     firstaccess=\relax,
10479     firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10480     text={\the\glsshorttok},%
10481     textaccess={\the\glslongtok},%
10482     plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10483     symbol={\noexpand\@glo@text},%
10484     symbolaccess={\noexpand\@glo@textaccess},%
10485     symbolplural={\noexpand\@glo@plural},%
10486     firstpluralaccess=\relax,
10487     textaccess={\noexpand\@glo@shortaccess},%
10488     \the\glskeylisttok}%
10489 }%
10490 \let\@org@gls@assign@firstpl\gls@assign@firstpl
10491 \let\@org@gls@assign@plural\gls@assign@plural
10492 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
10493 \def\gls@assign@firstpl##1##2{%
10494   \@@gls@expand@field{##1}{firstpl}{##2}%
10495 }%
10496 \def\gls@assign@plural##1##2{%
10497   \@@gls@expand@field{##1}{plural}{##2}%
10498 }%
10499 \def\gls@assign@symbolplural##1##2{%
10500   \@@gls@expand@field{##1}{symbolplural}{##2}%
10501 }%
10502 \do@newglossaryentry
10503 \let\gls@assign@firstpl\@org@gls@assign@firstpl
10504 \let\gls@assign@plural\@org@gls@assign@plural
10505 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
10506 }

```

otnoteNewAcronymDef

```

10507 \renewcommand*{\FootnoteNewAcronymDef}{%
10508   \edef\@do@newglossaryentry{%
10509     \noexpand\newglossaryentry{\the\glslabeltok}%
10510     {%
10511       type=\acronymtype,%
10512       name={\noexpand\acronymfont{\the\glsshorttok}},%
10513       sort={\the\glsshorttok},%
10514       text={\the\glsshorttok},%
10515       textaccess={\the\glslongtok},%
10516       access={\noexpand\@glo@textaccess},%

```

```

10517 plural={\the\glssshorttok\noexpand\acrpluralsuffix},%
10518 short={\the\glssshorttok},%
10519 shortplural={\the\glssshorttok\noexpand\acrpluralsuffix},%
10520 long={\the\glslongtok},%
10521 longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10522 description={\the\glslongtok},%
10523 descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10524 \the\glskeylisttok
10525 }%
10526 }%
10527 \let\@org@gls@assign@plural\gls@assign@plural
10528 \let\@org@gls@assign@firstpl\gls@assign@firstpl
10529 \let\@org@gls@assign@descplural\gls@assign@descplural
10530 \def\gls@assign@firstpl##1##2{%
10531   \@@gls@expand@field{##1}{firstpl}{##2}%
10532 }%
10533 \def\gls@assign@plural##1##2{%
10534   \@@gls@expand@field{##1}{plural}{##2}%
10535 }%
10536 \def\gls@assign@descplural##1##2{%
10537   \@@gls@expand@field{##1}{descplural}{##2}%
10538 }%
10539 \@do@newglossaryentry
10540 \let\gls@assign@plural\@org@gls@assign@plural
10541 \let\gls@assign@firstpl\@org@gls@assign@firstpl
10542 \let\gls@assign@descplural\@org@gls@assign@descplural
10543 }

```

\SmallNewAcronymDef

```

10544 \renewcommand*{\SmallNewAcronymDef}{%
10545   \edef\@do@newglossaryentry{%
10546     \noexpand\newglossaryentry{\the\glslabeltok}%
10547     {%
10548       type=\acronymtype,%
10549       name={\noexpand\acronymfont{\the\glssshorttok}},%
10550       access={\noexpand\@glo@symbolaccess},%
10551       sort={\the\glssshorttok},%
10552       short={\the\glssshorttok},%
10553       shortplural={\the\glssshorttok\noexpand\acrpluralsuffix},%
10554       shortaccess={\the\glslongtok},%
10555       long={\the\glslongtok},%
10556       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10557       text={\noexpand\@glo@short},%
10558       textaccess={\noexpand\@glo@shortaccess},%
10559       plural={\noexpand\@glo@shortpl},%
10560       first={\the\glslongtok},%
10561       firstaccess=\relax,%
10562       firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10563       description={\noexpand\@glo@first},%

```

```

10564     descriptionplural={\noexpand\@glo@firstplural},%
10565     symbol={\the\glsshorttok},%
10566     symbolaccess={\the\glslongtok},%
10567     symbolplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10568     \the\glskeylisttok
10569   }%
10570 }%
10571 \let\@org@gls@assign@firstpl\gls@assign@firstpl
10572 \let\@org@gls@assign@plural\gls@assign@plural
10573 \let\@org@gls@assign@descplural\gls@assign@descplural
10574 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
10575 \def\gls@assign@firstpl##1##2{%
10576   \@@gls@expand@field{##1}{firstpl}{##2}%
10577 }%
10578 \def\gls@assign@plural##1##2{%
10579   \@@gls@expand@field{##1}{plural}{##2}%
10580 }%
10581 \def\gls@assign@descplural##1##2{%
10582   \@@gls@expand@field{##1}{descplural}{##2}%
10583 }%
10584 \def\gls@assign@symbolplural##1##2{%
10585   \@@gls@expand@field{##1}{symbolplural}{##2}%
10586 }%
10587 \do@newglossaryentry
10588 \let\gls@assign@firstpl\@org@gls@assign@firstpl
10589 \let\gls@assign@plural\@org@gls@assign@plural
10590 \let\gls@assign@descplural\@org@gls@assign@descplural
10591 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
10592 }

```

The following are kept for compatibility with versions before 3.0:

```

\glsshortaccesskey
10593   \newcommand*{\glsshortaccesskey}{\glsshortkey access}%

hortpluralaccesskey
10594   \newcommand*{\glsshortpluralaccesskey}{\glsshortpluralkey access}%

\glslongaccesskey
10595   \newcommand*{\glslongaccesskey}{\glslongkey access}%

longpluralaccesskey
10596   \newcommand*{\glslongpluralaccesskey}{\glslongpluralkey access}%

```

## 7.5 Debugging Commands

```

\showglongnameaccess
10597 \newcommand*{\showglongnameaccess}[1]{%
10598   \expandafter\show\curname glo@\glsdetoklabel{#1}@textaccess\endcsname
10599 }

```

```

\showglotextaccess
10600 \newcommand*{\showglotextaccess}[1]{%
10601   \expandafter\show\csname glo@\glsdetoklabel{#1}@textaccess\endcsname
10602 }

showglopluralaccess
10603 \newcommand*{\showglopluralaccess}[1]{%
10604   \expandafter\show\csname glo@\glsdetoklabel{#1}@pluralaccess\endcsname
10605 }

\showglofirstaccess
10606 \newcommand*{\showglofirstaccess}[1]{%
10607   \expandafter\show\csname glo@\glsdetoklabel{#1}@firstaccess\endcsname
10608 }

lofirstpluralaccess
10609 \newcommand*{\showglofirstpluralaccess}[1]{%
10610   \expandafter\show\csname glo@\glsdetoklabel{#1}@firstpluralaccess\endcsname
10611 }

showglosymbolaccess
10612 \newcommand*{\showglosymbolaccess}[1]{%
10613   \expandafter\show\csname glo@\glsdetoklabel{#1}@symbolaccess\endcsname
10614 }

osymbolpluralaccess
10615 \newcommand*{\showglosymbolpluralaccess}[1]{%
10616   \expandafter\show\csname glo@\glsdetoklabel{#1}@symbolpluralaccess\endcsname
10617 }

\showglodescaccess
10618 \newcommand*{\showglodescaccess}[1]{%
10619   \expandafter\show\csname glo@\glsdetoklabel{#1}@descaccess\endcsname
10620 }

glodescpluralaccess
10621 \newcommand*{\showglodescpluralaccess}[1]{%
10622   \expandafter\show\csname glo@\glsdetoklabel{#1}@descpluralaccess\endcsname
10623 }

\showgloshortaccess
10624 \newcommand*{\showgloshortaccess}[1]{%
10625   \expandafter\show\csname glo@\glsdetoklabel{#1}@shortaccess\endcsname
10626 }

loshortpluralaccess
10627 \newcommand*{\showgloshortpluralaccess}[1]{%
10628   \expandafter\show\csname glo@\glsdetoklabel{#1}@shortpluralaccess\endcsname
10629 }

```

```

\showglolongaccess
10630 \newcommand*{\showglolongaccess}[1]{%
10631   \expandafter\show\csname glo@glstdetoklabel{#1}@longaccess\endcsname
10632 }

glolongpluralaccess
10633 \newcommand*{\showglolongpluralaccess}[1]{%
10634   \expandafter\show\csname glo@glstdetoklabel{#1}@longpluralaccess\endcsname
10635 }

```

## 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.

```

10636 \NeedsTeXFormat{LaTeX2e}
10637 \ProvidesPackage{glossaries-babel}[2013/11/14 v4.0 (NLCT)]

English:
10638 \@ifundefined{captionsenglish}{-}{%
10639   \addto\captionsenglish{%
10640     \renewcommand*{\glossaryname}{Glossary}%
10641     \renewcommand*{\acronymname}{Acronyms}%
10642     \renewcommand*{\entryname}{Notation}%
10643     \renewcommand*{\descriptionname}{Description}%
10644     \renewcommand*{\symbolname}{Symbol}%
10645     \renewcommand*{\pagelistname}{Page List}%
10646     \renewcommand*{\glssymbolsgroupname}{Symbols}%
10647     \renewcommand*{\glsnumbersgroupname}{Numbers}%
10648 }%
10649 }
10650 \@ifundefined{captionsamerican}{-}{%
10651   \addto\captionsamerican{%
10652     \renewcommand*{\glossaryname}{Glossary}%
10653     \renewcommand*{\acronymname}{Acronyms}%
10654     \renewcommand*{\entryname}{Notation}%
10655     \renewcommand*{\descriptionname}{Description}%
10656     \renewcommand*{\symbolname}{Symbol}%
10657     \renewcommand*{\pagelistname}{Page List}%
10658     \renewcommand*{\glssymbolsgroupname}{Symbols}%
10659     \renewcommand*{\glsnumbersgroupname}{Numbers}%
10660 }%
10661 }
10662 \@ifundefined{captionsaustralian}{-}{%

```



```

10663 \addto\captionsaustrian{%
10664 \renewcommand*{\glossaryname}{Glossary}%
10665 \renewcommand*{\acronymname}{Acronyms}%
10666 \renewcommand*{\entryname}{Notation}%
10667 \renewcommand*{\descriptionname}{Description}%
10668 \renewcommand*{\symbolname}{Symbol}%
10669 \renewcommand*{\pagelistname}{Page List}%
10670 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10671 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10672 }%
10673 }
10674 \@ifundefined{captionsbritish}{-}{%
10675 \addto\captionsbritish{%
10676 \renewcommand*{\glossaryname}{Glossary}%
10677 \renewcommand*{\acronymname}{Acronyms}%
10678 \renewcommand*{\entryname}{Notation}%
10679 \renewcommand*{\descriptionname}{Description}%
10680 \renewcommand*{\symbolname}{Symbol}%
10681 \renewcommand*{\pagelistname}{Page List}%
10682 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10683 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10684 }%
10685 \@ifundefined{captionscanadian}{-}{%
10686 \addto\captionscanadian{%
10687 \renewcommand*{\glossaryname}{Glossary}%
10688 \renewcommand*{\acronymname}{Acronyms}%
10689 \renewcommand*{\entryname}{Notation}%
10690 \renewcommand*{\descriptionname}{Description}%
10691 \renewcommand*{\symbolname}{Symbol}%
10692 \renewcommand*{\pagelistname}{Page List}%
10693 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10694 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10695 }%
10696 }
10697 \@ifundefined{captionsnewzealand}{-}{%
10698 \addto\captionnewzealand{%
10699 \renewcommand*{\glossaryname}{Glossary}%
10700 \renewcommand*{\acronymname}{Acronyms}%
10701 \renewcommand*{\entryname}{Notation}%
10702 \renewcommand*{\descriptionname}{Description}%
10703 \renewcommand*{\symbolname}{Symbol}%
10704 \renewcommand*{\pagelistname}{Page List}%
10705 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10706 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10707 }%
10708 }
10709 \@ifundefined{captionsUKenglish}{-}{%
10710 \addto\captionUKenglish{%
10711 \renewcommand*{\glossaryname}{Glossary}%

```

```

10712 \renewcommand*{\acronymname}{Acronyms}%
10713 \renewcommand*{\entryname}{Notation}%
10714 \renewcommand*{\descriptionname}{Description}%
10715 \renewcommand*{\symbolname}{Symbol}%
10716 \renewcommand*{\pagelistname}{Page List}%
10717 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10718 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10719 }%
10720 }
10721 \@ifundefined{captionsUSenglish}{-}{%
10722 \addto\captionsUSenglish{%
10723 \renewcommand*{\glossaryname}{Glossary}%
10724 \renewcommand*{\acronymname}{Acronyms}%
10725 \renewcommand*{\entryname}{Notation}%
10726 \renewcommand*{\descriptionname}{Description}%
10727 \renewcommand*{\symbolname}{Symbol}%
10728 \renewcommand*{\pagelistname}{Page List}%
10729 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10730 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10731 }%
10732 }

```

German (quite a few variations were suggested for German; I settled on the following):

```

10733 \@ifundefined{captionsgerman}{-}{%
10734 \addto\captionsgerman{%
10735 \renewcommand*{\glossaryname}{Glossar}%
10736 \renewcommand*{\acronymname}{Akronyme}%
10737 \renewcommand*{\entryname}{Bezeichnung}%
10738 \renewcommand*{\descriptionname}{Beschreibung}%
10739 \renewcommand*{\symbolname}{Symbol}%
10740 \renewcommand*{\pagelistname}{Seiten}%
10741 \renewcommand*{\glssymbolsgroupname}{Symbole}%
10742 \renewcommand*{\glsnumbersgroupname}{Zahlen}}
10743 }

```

ngerman is identical to German:

```

10744 \@ifundefined{captionsgerman}{-}{%
10745 \addto\captionsgerman{%
10746 \renewcommand*{\glossaryname}{Glossar}%
10747 \renewcommand*{\acronymname}{Akronyme}%
10748 \renewcommand*{\entryname}{Bezeichnung}%
10749 \renewcommand*{\descriptionname}{Beschreibung}%
10750 \renewcommand*{\symbolname}{Symbol}%
10751 \renewcommand*{\pagelistname}{Seiten}%
10752 \renewcommand*{\glssymbolsgroupname}{Symbole}%
10753 \renewcommand*{\glsnumbersgroupname}{Zahlen}}
10754 }

```

Italian:

```

10755 \@ifundefined{captionsitalian}{-}{%

```

```

10756 \addto\captionssitalian{%
10757   \renewcommand*{\glossaryname}{Glossario}%
10758   \renewcommand*{\acronymname}{Acronimi}%
10759   \renewcommand*{\entryname}{Nomenclatura}%
10760   \renewcommand*{\descriptionname}{Descrizione}%
10761   \renewcommand*{\symbolname}{Simbolo}%
10762   \renewcommand*{\pagelistname}{Elenco delle pagine}%
10763   \renewcommand*{\glssymbolsgroupname}{Simboli}%
10764   \renewcommand*{\glsnumbersgroupname}{Numeri}}
10765 }

```

Dutch:

```

10766 \@ifundefined{captionssdutch}{-}{%
10767   \addto\captionssdutch{%
10768     \renewcommand*{\glossaryname}{Woordenlijst}%
10769     \renewcommand*{\acronymname}{Acroniemen}%
10770     \renewcommand*{\entryname}{Benaming}%
10771     \renewcommand*{\descriptionname}{Beschrijving}%
10772     \renewcommand*{\symbolname}{Symbool}%
10773     \renewcommand*{\pagelistname}{Pagina's}%
10774     \renewcommand*{\glssymbolsgroupname}{Symbolen}%
10775     \renewcommand*{\glsnumbersgroupname}{Cijfers}}
10776 }

```

Spanish:

```

10777 \@ifundefined{captionssspanish}{-}{%
10778   \addto\captionssspanish{%
10779     \renewcommand*{\glossaryname}{Glosario}%
10780     \renewcommand*{\acronymname}{Siglas}%
10781     \renewcommand*{\entryname}{Entrada}%
10782     \renewcommand*{\descriptionname}{Descripci\'on}%
10783     \renewcommand*{\symbolname}{S\'mbolo}%
10784     \renewcommand*{\pagelistname}{Lista de p\'aginas}%
10785     \renewcommand*{\glssymbolsgroupname}{S\'mbolos}%
10786     \renewcommand*{\glsnumbersgroupname}{N\'umeros}}
10787 }

```

French:

```

10788 \@ifundefined{captionssfrench}{-}{%
10789   \addto\captionssfrench{%
10790     \renewcommand*{\glossaryname}{Glossaire}%
10791     \renewcommand*{\acronymname}{Acronymes}%
10792     \renewcommand*{\entryname}{Termes}%
10793     \renewcommand*{\descriptionname}{Description}%
10794     \renewcommand*{\symbolname}{Symbole}%
10795     \renewcommand*{\pagelistname}{Pages}%
10796     \renewcommand*{\glssymbolsgroupname}{Symboles}%
10797     \renewcommand*{\glsnumbersgroupname}{Nombres}}
10798 }
10799 \@ifundefined{captionssfrenchb}{-}{%
10800   \addto\captionssfrenchb{%

```

```

10801 \renewcommand*{\glossaryname}{Glossaire}%
10802 \renewcommand*{\acronymname}{Acronymes}%
10803 \renewcommand*{\entryname}{Terme}%
10804 \renewcommand*{\descriptionname}{Description}%
10805 \renewcommand*{\symbolname}{Symbole}%
10806 \renewcommand*{\pagelistname}{Pages}%
10807 \renewcommand*{\glssymbolsgroupname}{Symboles}%
10808 \renewcommand*{\glslnumbersgroupname}{Nombres}}
10809 }
10810 \@ifundefined{captionsfrancais}{}{%
10811 \addto\captionsfrancais{%
10812 \renewcommand*{\glossaryname}{Glossaire}%
10813 \renewcommand*{\acronymname}{Acronymes}%
10814 \renewcommand*{\entryname}{Terme}%
10815 \renewcommand*{\descriptionname}{Description}%
10816 \renewcommand*{\symbolname}{Symbole}%
10817 \renewcommand*{\pagelistname}{Pages}%
10818 \renewcommand*{\glssymbolsgroupname}{Symboles}%
10819 \renewcommand*{\glslnumbersgroupname}{Nombres}}
10820 }

```

Danish:

```

10821 \@ifundefined{captionsdanish}{}{%
10822 \addto\captionsdanish{%
10823 \renewcommand*{\glossaryname}{Ordliste}%
10824 \renewcommand*{\acronymname}{Akronymer}%
10825 \renewcommand*{\entryname}{Symbolforklaring}%
10826 \renewcommand*{\descriptionname}{Beskrivelse}%
10827 \renewcommand*{\symbolname}{Symbol}%
10828 \renewcommand*{\pagelistname}{Side}%
10829 \renewcommand*{\glssymbolsgroupname}{Symboler}%
10830 \renewcommand*{\glslnumbersgroupname}{Tal}}
10831 }

```

Irish:

```

10832 \@ifundefined{captionsirish}{}{%
10833 \addto\captionsirish{%
10834 \renewcommand*{\glossaryname}{Gluais}%
10835 \renewcommand*{\acronymname}{Acrainmneacha}%

```

wasn't sure whether to go for Nóta (Note), Ciall ('Meaning', 'sense') or Brí ('Meaning'). In the end I chose Ciall.

```

10836 \renewcommand*{\entryname}{Ciall}%
10837 \renewcommand*{\descriptionname}{Tuairisc}%

```

Again, not sure whether to use Comhartha/Comharthaí or Siombail/Siombaile, so have chosen the former.

```

10838 \renewcommand*{\symbolname}{Comhartha}%
10839 \renewcommand*{\glssymbolsgroupname}{Comhartha'\{i}}%
10840 \renewcommand*{\pagelistname}{Leathanaigh}%
10841 \renewcommand*{\glslnumbersgroupname}{Uimhreacha}}

```

10842 }

#### Hungarian:

```
10843 \@ifundefined{captionsmagyar}{%{}%
10844   \addto\captionsmagyar{%
10845     \renewcommand*{\glossaryname}{Sz\`ojegyz\`ek}%
10846     \renewcommand*{\acronymname}{Bet\H uszavak}%
10847     \renewcommand*{\entryname}{Kifejez\`es}%
10848     \renewcommand*{\descriptionname}{Magyar\`azat}%
10849     \renewcommand*{\symbolname}{Jel\"ol\`es}%
10850     \renewcommand*{\pagelistname}{Oldalsz\`am}%
10851     \renewcommand*{\glssymbolsgroupname}{Jelek}%
10852     \renewcommand*{\glsnumbersgroupname}{Sz\`amjegyek}%
10853   }
10854 }
10855 \@ifundefined{captionshungarian}{%{}%
10856   \addto\captionshungarian{%
10857     \renewcommand*{\glossaryname}{Sz\`ojegyz\`ek}%
10858     \renewcommand*{\acronymname}{Bet\H uszavak}%
10859     \renewcommand*{\entryname}{Kifejez\`es}%
10860     \renewcommand*{\descriptionname}{Magyar\`azat}%
10861     \renewcommand*{\symbolname}{Jel\"ol\`es}%
10862     \renewcommand*{\pagelistname}{Oldalsz\`am}%
10863     \renewcommand*{\glssymbolsgroupname}{Jelek}%
10864     \renewcommand*{\glsnumbersgroupname}{Sz\`amjegyek}%
10865   }
10866 }
```

#### Polish

```
10867 \@ifundefined{captionspolish}{%{}%
10868   \addto\captionspolish{%
10869     \renewcommand*{\glossaryname}{S{\l}ownik termin\`ow}%
10870     \renewcommand*{\acronymname}{Skr\`ot}%
10871     \renewcommand*{\entryname}{Termin}%
10872     \renewcommand*{\descriptionname}{Opis}%
10873     \renewcommand*{\symbolname}{Symbol}%
10874     \renewcommand*{\pagelistname}{Strony}%
10875     \renewcommand*{\glssymbolsgroupname}{Symbole}%
10876     \renewcommand*{\glsnumbersgroupname}{Liczby}}
10877 }
```

#### Brazilian

```
10878 \@ifundefined{captionsbrazil}{%{}%
10879   \addto\captionsbrazil{%
10880     \renewcommand*{\glossaryname}{Gloss\`ario}%
10881     \renewcommand*{\acronymname}{Siglas}%
10882     \renewcommand*{\entryname}{Nota\c c\~ao}%
10883     \renewcommand*{\descriptionname}{Descri\c c\~ao}%
10884     \renewcommand*{\symbolname}{S\`imbolo}%
10885     \renewcommand*{\pagelistname}{Lista de P\`aginas}%
10886     \renewcommand*{\glssymbolsgroupname}{S\`imbolos}%

```

```

10887 \renewcommand*{\glsnumbersgroupname}{N\,umeros}%
10888 }%
10889 }

```

## 8.2 Polyglossia Captions

```

10890 \NeedsTeXFormat{LaTeX2e}
10891 \ProvidesPackage{glossaries-polyglossia}[2013/11/14 v4.0 (NLCT)]

```

English:

```

10892 \@ifundefined{captionsenglish}{}{%
10893 \expandafter\toks@\expandafter{\captionsenglish
10894 \renewcommand*{\glossaryname}{\textenglish{Glossary}}%
10895 \renewcommand*{\acronymname}{\textenglish{Acronyms}}%
10896 \renewcommand*{\entryname}{\textenglish{Notation}}%
10897 \renewcommand*{\descriptionname}{\textenglish{Description}}%
10898 \renewcommand*{\symbolname}{\textenglish{Symbol}}%
10899 \renewcommand*{\pagelistname}{\textenglish{Page List}}%
10900 \renewcommand*{\glssymbolsgroupname}{\textenglish{Symbols}}%
10901 \renewcommand*{\glsnumbersgroupname}{\textenglish{Numbers}}%
10902 }%
10903 \edef\captionsenglish{\the\toks@}%
10904 }

```

German:

```

10905 \@ifundefined{captionsgerman}{}{%
10906 \expandafter\toks@\expandafter{\captionsgerman
10907 \renewcommand*{\glossaryname}{\textgerman{Glossar}}%
10908 \renewcommand*{\acronymname}{\textgerman{Akronyme}}%
10909 \renewcommand*{\entryname}{\textgerman{Bezeichnung}}%
10910 \renewcommand*{\descriptionname}{\textgerman{Beschreibung}}%
10911 \renewcommand*{\symbolname}{\textgerman{Symbol}}%
10912 \renewcommand*{\pagelistname}{\textgerman{Seiten}}%
10913 \renewcommand*{\glssymbolsgroupname}{\textgerman{Symbole}}%
10914 \renewcommand*{\glsnumbersgroupname}{\textgerman{Zahlen}}%
10915 }%
10916 \edef\captionsgerman{\the\toks@}%
10917 }

```

Italian:

```

10918 \@ifundefined{captionsitalian}{}{%
10919 \expandafter\toks@\expandafter{\captionsitalian
10920 \renewcommand*{\glossaryname}{\textitalian{Glossario}}%
10921 \renewcommand*{\acronymname}{\textitalian{Acronimi}}%
10922 \renewcommand*{\entryname}{\textitalian{Nomenclatura}}%
10923 \renewcommand*{\descriptionname}{\textitalian{Descrizione}}%
10924 \renewcommand*{\symbolname}{\textitalian{Simbolo}}%
10925 \renewcommand*{\pagelistname}{\textitalian{Elenco delle pagine}}%
10926 \renewcommand*{\glssymbolsgroupname}{\textitalian{Simboli}}%
10927 \renewcommand*{\glsnumbersgroupname}{\textitalian{Numeri}}%
10928 }%

```

```

10929 \edef\captionsitalian{\the\toks@}%
10930 }

```

#### Dutch:

```

10931 \@ifundefined{captionsdutch}{}{%
10932 \expandafter\toks@\expandafter{\captionsdutch
10933 \renewcommand*{\glossaryname}{\textdutch{Woordenlijst}}%
10934 \renewcommand*{\acronymname}{\textdutch{Acroniemen}}%
10935 \renewcommand*{\entryname}{\textdutch{Benaming}}%
10936 \renewcommand*{\descriptionname}{\textdutch{Beschrijving}}%
10937 \renewcommand*{\symbolname}{\textdutch{Symbool}}%
10938 \renewcommand*{\pagelistname}{\textdutch{Pagina's}}%
10939 \renewcommand*{\glssymbolsgroupname}{\textdutch{Symbolen}}%
10940 \renewcommand*{\glsnumbersgroupname}{\textdutch{Cijfers}}%
10941 }%
10942 \edef\captionsdutch{\the\toks@}%
10943 }

```

#### Spanish:

```

10944 \@ifundefined{captionsspanish}{}{%
10945 \expandafter\toks@\expandafter{\captionsspanish
10946 \renewcommand*{\glossaryname}{\textspanish{Glosario}}%
10947 \renewcommand*{\acronymname}{\textspanish{Siglas}}%
10948 \renewcommand*{\entryname}{\textspanish{Entrada}}%
10949 \renewcommand*{\descriptionname}{\textspanish{Descripci'on}}%
10950 \renewcommand*{\symbolname}{\textspanish{S'\i mbolo}}%
10951 \renewcommand*{\pagelistname}{\textspanish{Lista de p'aginas}}%
10952 \renewcommand*{\glssymbolsgroupname}{\textspanish{S'\i mbolos}}%
10953 \renewcommand*{\glsnumbersgroupname}{\textspanish{N'umeros}}%
10954 }%
10955 \edef\captionsspanish{\the\toks@}%
10956 }

```

#### French:

```

10957 \@ifundefined{captionsfrench}{}{%
10958 \expandafter\toks@\expandafter{\captionsfrench
10959 \renewcommand*{\glossaryname}{\textfrench{Glossaire}}%
10960 \renewcommand*{\acronymname}{\textfrench{Acronymes}}%
10961 \renewcommand*{\entryname}{\textfrench{Terme}}%
10962 \renewcommand*{\descriptionname}{\textfrench{Description}}%
10963 \renewcommand*{\symbolname}{\textfrench{Symbole}}%
10964 \renewcommand*{\pagelistname}{\textfrench{Pages}}%
10965 \renewcommand*{\glssymbolsgroupname}{\textfrench{Symboles}}%
10966 \renewcommand*{\glsnumbersgroupname}{\textfrench{Nombres}}%
10967 }%
10968 \edef\captionsfrench{\the\toks@}%
10969 }

```

#### Danish:

```

10970 \@ifundefined{captionsdanish}{}{%
10971 \expandafter\toks@\expandafter{\captionsdanish
10972 \renewcommand*{\glossaryname}{\textdanish{Ordliste}}%

```

```

10973 \renewcommand*{\acronymname}{\textdanish{Akronymer}}}%
10974 \renewcommand*{\entryname}{\textdanish{Symbolforklaring}}}%
10975 \renewcommand*{\descriptionname}{\textdanish{Beskrivelse}}}%
10976 \renewcommand*{\symbolname}{\textdanish{Symbol}}}%
10977 \renewcommand*{\pagelistname}{\textdanish{Side}}}%
10978 \renewcommand*{\glssymbolsgroupname}{\textdanish{Symboler}}}%
10979 \renewcommand*{\glsnumbersgroupname}{\textdanish{Tal}}}%
10980 }%
10981 \edef\captionsdanish{\the\toks@}%
10982 }

Irish:
10983 \@ifundefined{captionsirish}{}{%
10984 \expandafter\toks@\expandafter{\captionsirish
10985 \renewcommand*{\glossaryname}{\textirish{Gluais}}}%
10986 \renewcommand*{\acronymname}{\textirish{Acrainmneacha}}}%
10987 \renewcommand*{\entryname}{\textirish{Ciall}}}%
10988 \renewcommand*{\descriptionname}{\textirish{Tuirisc}}}%
10989 \renewcommand*{\symbolname}{\textirish{Comhartha}}}%
10990 \renewcommand*{\glssymbolsgroupname}{\textirish{Comhartha\`{\i}}}%
10991 \renewcommand*{\pagelistname}{\textirish{Leathanaigh}}}%
10992 \renewcommand*{\glsnumbersgroupname}{\textirish{Uimhreacha}}}%
10993 }%
10994 \edef\captionsirish{\the\toks@}%
10995 }

Hungarian:
10996 \@ifundefined{captionsmagyar}{}{%
10997 \expandafter\toks@\expandafter{\captionsmagyar
10998 \renewcommand*{\glossaryname}{\textmagyar{Sz\`ojegyz\`ek}}}%
10999 \renewcommand*{\acronymname}{\textmagyar{Bet\H uszavak}}}%
11000 \renewcommand*{\entryname}{\textmagyar{Kifejez\`es}}}%
11001 \renewcommand*{\descriptionname}{\textmagyar{Magyar\`azat}}}%
11002 \renewcommand*{\symbolname}{\textmagyar{Jel\`ol\`es}}}%
11003 \renewcommand*{\pagelistname}{\textmagyar{Oldalsz\`am}}}%
11004 \renewcommand*{\glssymbolsgroupname}{\textmagyar{Jelek}}}%
11005 \renewcommand*{\glsnumbersgroupname}{\textmagyar{Sz\`amjegyek}}}%
11006 }%
11007 \edef\captionsmagyar{\the\toks@}%
11008 }

Polish
11009 \@ifundefined{captionspolish}{}{%
11010 \expandafter\toks@\expandafter{\captionspolish
11011 \renewcommand*{\glossaryname}{\textpolish{S{\l}ownik termin\`ow}}}%
11012 \renewcommand*{\acronymname}{\textpolish{Skr\`ot}}}%
11013 \renewcommand*{\entryname}{\textpolish{Termin}}}%
11014 \renewcommand*{\descriptionname}{\textpolish{Opis}}}%
11015 \renewcommand*{\symbolname}{\textpolish{Symbol}}}%
11016 \renewcommand*{\pagelistname}{\textpolish{Strony}}}%
11017 \renewcommand*{\glssymbolsgroupname}{\textpolish{Symbole}}}%

```



```

11018 \renewcommand*{\glsnumbersgroupname}{\textpolish{Liczby}}%
11019 }%
11020 \edef\captionspolish{\the\toks@}%
11021 }

```

### Portugues

```

11022 \@ifundefined{captionoportuges}{}{%
11023 \expandafter\toks@\expandafter{\captionoportuges
11024 \renewcommand*{\glossaryname}{\textportuges{Gloss\'ario}}%
11025 \renewcommand*{\acronymname}{\textportuges{Siglas}}%
11026 \renewcommand*{\entryname}{\textportuges{Nota\c c\~ao}}%
11027 \renewcommand*{\descriptionname}{\textportuges{Descri\c c\~ao}}%
11028 \renewcommand*{\symbolname}{\textportuges{S\'imbolo}}%
11029 \renewcommand*{\pagelistname}{\textportuges{Lista de P\'aginas}}%
11030 \renewcommand*{\glssymbolsgroupname}{\textportuges{S\'imbolos}}%
11031 \renewcommand*{\glsnumbersgroupname}{\textportuges{N\'umeros}}%
11032 }%
11033 \edef\captionoportuges{\the\toks@}%
11034 }

```

## 8.3 Brazilian Dictionary

This is a dictionary file provided by Thiago de Melo for use with the package.

```

11035 \ProvidesDictionary{glossaries-dictionary}{Brazilian}

```

Provide Brazilian translations:

```

11036 \providetranslation{Glossary}{Gloss\'ario}
11037 \providetranslation{Acronyms}{Siglas}
11038 \providetranslation{Notation (glossaries)}{Nota\c c\~ao}
11039 \providetranslation{Description (glossaries)}{Descri\c c\~ao}
11040 \providetranslation{Symbol (glossaries)}{S\'imbolo}
11041 \providetranslation{Page List (glossaries)}{Lista de P\'aginas}
11042 \providetranslation{Symbols (glossaries)}{S\'imbolos}
11043 \providetranslation{Numbers (glossaries)}{N\'umeros}

```

## 8.4 Danish Dictionary

This is a dictionary file provided for use with the package.

```

11044 \ProvidesDictionary{glossaries-dictionary}{Danish}

```

Provide Danish translations:

```

11045 \providetranslation{Glossary}{Ordliste}
11046 \providetranslation{Acronyms}{Akronymer}
11047 \providetranslation{Notation (glossaries)}{Symbolforklaring}
11048 \providetranslation{Description (glossaries)}{Beskrivelse}
11049 \providetranslation{Symbol (glossaries)}{Symbol}
11050 \providetranslation{Page List (glossaries)}{Side}
11051 \providetranslation{Symbols (glossaries)}{Symboler}
11052 \providetranslation{Numbers (glossaries)}{Tal}

```

## 8.5 Dutch Dictionary

This is a dictionary file provided for use with the package.

```
11053 \ProvidesDictionary{glossaries-dictionary}{Dutch}
```

Provide Dutch translations:

```
11054 \providetranslation{Glossary}{Woordenlijst}
11055 \providetranslation{Acronyms}{Acroniemen}
11056 \providetranslation{Notation (glossaries)}{Benaming}
11057 \providetranslation{Description (glossaries)}{Beschrijving}
11058 \providetranslation{Symbol (glossaries)}{Symbool}
11059 \providetranslation{Page List (glossaries)}{Pagina's}
11060 \providetranslation{Symbols (glossaries)}{Symbolen}
11061 \providetranslation{Numbers (glossaries)}{Cijfers}
```

## 8.6 English Dictionary

This is a dictionary file provided for use with the package.

```
11062 \ProvidesDictionary{glossaries-dictionary}{English}
```

Provide English translations:

```
11063 \providetranslation{Glossary}{Glossary}
11064 \providetranslation{Acronyms}{Acronyms}
11065 \providetranslation{Notation (glossaries)}{Notation}
11066 \providetranslation{Description (glossaries)}{Description}
11067 \providetranslation{Symbol (glossaries)}{Symbol}
11068 \providetranslation{Page List (glossaries)}{Page List}
11069 \providetranslation{Symbols (glossaries)}{Symbols}
11070 \providetranslation{Numbers (glossaries)}{Numbers}
```

## 8.7 French Dictionary

This is a dictionary file provided for use with the package.

```
11071 \ProvidesDictionary{glossaries-dictionary}{French}
```

Provide French translations:

```
11072 \providetranslation{Glossary}{Glossaire}
11073 \providetranslation{Acronyms}{Acronymes}
11074 \providetranslation{Notation (glossaries)}{Terme}
11075 \providetranslation{Description (glossaries)}{Description}
11076 \providetranslation{Symbol (glossaries)}{Symbole}
11077 \providetranslation{Page List (glossaries)}{Pages}
11078 \providetranslation{Symbols (glossaries)}{Symboles}
11079 \providetranslation{Numbers (glossaries)}{Nombres}
```

## 8.8 German Dictionary

This is a dictionary file provided for use with the package.

```
11080 \ProvidesDictionary{glossaries-dictionary}{German}
```

Provide German translations (quite a few variations were suggested for German; I settled on the following):

```
11081 \providetranslation{Glossary}{Glossar}
11082 \providetranslation{Acronyms}{Akronyme}
11083 \providetranslation{Notation (glossaries)}{Bezeichnung}
11084 \providetranslation{Description (glossaries)}{Beschreibung}
11085 \providetranslation{Symbol (glossaries)}{Symbol}
11086 \providetranslation{Page List (glossaries)}{Seiten}
11087 \providetranslation{Symbols (glossaries)}{Symbole}
11088 \providetranslation{Numbers (glossaries)}{Zahlen}
```

## 8.9 Irish Dictionary

This is a dictionary file provided for use with the package.

```
11089 \ProvidesDictionary{glossaries-dictionary}{Irish}
```

Provide Irish translations:

```
11090 \providetranslation{Glossary}{Gluais}
11091 \providetranslation{Acronyms}{Acrainmneacha}
11092 \providetranslation{Notation (glossaries)}{Ciall}
11093 \providetranslation{Description (glossaries)}{Tuairisc}
11094 \providetranslation{Symbol (glossaries)}{Comhartha}
11095 \providetranslation{Page List (glossaries)}{Leathanaigh}
11096 \providetranslation{Symbols (glossaries)}{Comhartha'\{i}}
11097 \providetranslation{Numbers (glossaries)}{Uimhreacha}
```

## 8.10 Italian Dictionary

This is a dictionary file provided for use with the package.

```
11098 \ProvidesDictionary{glossaries-dictionary}{Italian}
```

Provide Italian translations:

```
11099 \providetranslation{Glossary}{Glossario}
11100 \providetranslation{Acronyms}{Acronimi}
11101 \providetranslation{Notation (glossaries)}{Nomenclatura}
11102 \providetranslation{Description (glossaries)}{Descrizione}
11103 \providetranslation{Symbol (glossaries)}{Simbolo}
11104 \providetranslation{Page List (glossaries)}{Elenco delle pagine}
11105 \providetranslation{Symbols (glossaries)}{Simboli}
11106 \providetranslation{Numbers (glossaries)}{Numeri}
```

## 8.11 Magyar Dictionary

This is a dictionary file provided for use with the package.

```
11107 \ProvidesDictionary{glossaries-dictionary}{Magyar}
```

Provide translations:

```
11108 \providetranslation{Glossary}{Sz\'ojegyz\'ek}
11109 \providetranslation{Acronyms}{Bet\H uszavak}
```

```

11110 \providetranslation{Notation (glossaries)}{Kifejez\`es}
11111 \providetranslation{Description (glossaries)}{Magyar\`azat}
11112 \providetranslation{Symbol (glossaries)}{Jel\`ol\`es}
11113 \providetranslation{Page List (glossaries)}{Oldalsz\`am}
11114 \providetranslation{Symbols (glossaries)}{Jelek}
11115 \providetranslation{Numbers (glossaries)}{Sz\`amjegyek}

```

## 8.12 Polish Dictionary

This is a dictionary file provided for use with the package.

```

11116 \ProvidesDictionary{glossaries-dictionary}{Polish}

```

Provide Polish translations:

```

11117 \providetranslation{Glossary}{S\`ownik termin\`ow}
11118 \providetranslation{Acronyms}{Skr\`ot}
11119 \providetranslation{Notation (glossaries)}{Termin}
11120 \providetranslation{Description (glossaries)}{Opis}
11121 \providetranslation{Symbol (glossaries)}{Symbol}
11122 \providetranslation{Page List (glossaries)}{Strony}
11123 \providetranslation{Symbols (glossaries)}{Symbole}
11124 \providetranslation{Numbers (glossaries)}{Liczby}

```

## 8.13 Serbian Dictionary

This dictionary was provided by Zoran Filipovic.

```

11125 \ProvidesDictionary{glossaries-dictionary}{Serbian}
11126 \providetranslation{Glossary}{Mali re\`v cnik}
11127 \providetranslation{Acronyms}{Skra\` cenice}
11128 \providetranslation{Notation (glossaries)}{Oznaka}
11129 \providetranslation{Description (glossaries)}{Opis}
11130 \providetranslation{Symbol (glossaries)}{Simbol}
11131 \providetranslation{Page List (glossaries)}{Stranica}
11132 \providetranslation{Symbols (glossaries)}{Simboli}
11133 \providetranslation{Numbers (glossaries)}{Brojevi}

```

## 8.14 Spanish Dictionary

This is a dictionary file provided for use with the package.

```

11134 \ProvidesDictionary{glossaries-dictionary}{Spanish}

```

Provide Spanish translations:

```

11135 \providetranslation{Glossary}{Glosario}
11136 \providetranslation{Acronyms}{Siglas}
11137 \providetranslation{Notation (glossaries)}{Entrada}
11138 \providetranslation{Description (glossaries)}{Descripci\`on}
11139 \providetranslation{Symbol (glossaries)}{S\`{\i}mbolo}
11140 \providetranslation{Page List (glossaries)}{Lista de p\`aginas}
11141 \providetranslation{Symbols (glossaries)}{S\`{\i}mbolos}
11142 \providetranslation{Numbers (glossaries)}{N\`umeros}

```

## Glossary

`makeindex` An indexing application. 10, 23, 24

`xindy` An flexible indexing application with multilingual support written in Perl. 10, 23, 24

## Change History

??		<code>\capitalisewords</code> : made robust	
	super: fixed typo in <code>\subglossentry</code>	.....	242
	( <code>\glossentrydesc</code> )	.....	268
1.01	General: Added range facility in		
	format key	.....	96
	<code>\writeist</code> : Added spaces after		
	<code>\delimN</code> and <code>\delimR</code> in ist		
	file	.....	142
1.03	<code>\makefirstuc</code> : changed 'pro-		
	ected@edef to 'def	.....	241
1.04	General: Added <code>\glstextformat</code>		81
1.05	<code>\glossarysection</code> : added		
	<code>\mkboth</code> to <code>\glossarysection</code>	.....	36
	<code>\gls@defglossaryentry</code> :		
	Changed the default value of		
	the sort key to just the value of		
	the name key	.....	71
	<code>\glsmakefirstuc</code> : new	.....	242
1.06	General: now requires <code>etoolbox</code>		240
	<code>\capitalisewords</code> : new	.....	242
	<code>\xcapitalisewords</code> : new	....	243
1.07	<code>\gls@link</code> : fixed bug caused by		
	<code>\theglsentrycounter</code> set-		
	ting the page number too soon		94
	<code>\glsadd</code> : fixed bug caused by		
	<code>\theglsentrycounter</code> set-		
	ting the page number too soon	.....	140
1.08	General: Added babel support	...	30
	<code>listgroup</code> : changed <code>listgroup</code>		
	style to use <code>\glsgetgrouptitle</code>	.....	249
	<code>altlistgroup</code> : changed <code>al-</code>		
	<code>tlistgroup</code> style to use		
	<code>\glsgetgrouptitle</code>	.....	250
	<code>\makefirstuc</code> : made robust	...	240
1.09	<code>\mfu@nocaplist</code> : new	.....	243
	<code>\capitalisewords</code> : added check		
	for words that shouldn't be		
	capitalised	.....	242
	<code>\gMFUnocap</code> : new	.....	243
	<code>\mfu@checkword</code> : new	.....	242
	<code>\MFUclear</code> : new	.....	243
1.1	<code>\glossarysection</code> : numbered		
	sections and auto label added		37
	<code>\gls@tmpb</code> : changed <code>\toksdef</code>		
	to <code>\newtoks</code>	.....	99
	<code>\gls@toc</code> : numberline added	..	38
	<code>\p@glossarysection</code> : num-		
	bered sections and auto label		
	added	.....	37
	General: Added support for trans-		
	lator package	.....	31
	<code>amsgen</code> now loaded ( <code>\new@ifnextchar</code>		
	needed)	.....	4
	<code>translate</code> : <code>translate</code> option		
	added	.....	21
	<code>\setglossarysection</code> : new	...	37
	<code>numberedsection</code> : numbered-		
	section package option added	..	6
	<code>numberline</code> : <code>numberline</code> option		
	added	.....	5

1.12	Removed restriction on only using \newglossaryentry in the preamble . . . . .	76
\@GLSpl: now uses \glsentrydescplural and \glsentrysymbolplural instead of \glsentrydesc and \glsentrysymbol . . . .	110	\newacronym: Removed restriction on only using \newacronym in the preamble 193
\@GLspl@: now uses \glsentrydescplural and \glsentrysymbolplural instead of \glsentrydesc and \glsentrysymbol . . . .	109	1.14 \gls@hypergroup: new . . . . . 245
\@GLspl@: now uses \glsentrydescplural and \glsentrysymbolplural instead of \glsentrydesc and \glsentrysymbol . . . .	109	General: added nonnumberlist key to \printglossary . . . . . 179
General: added check for \hypertarget separate to \hyperlink (memoir defines \hyperlink but not \hypertarget) . . . . .	105	added numberedsection key to \printglossary . . . . . 177
descriptionplural: new . . . . .	58	\firstacronymfont: new . . . . . 197
\gls@defglossaryentry: Changed default first plural to be first key with s appended (was text key with s appended) . . . . .	71	\glsautoprefix: new . . . . . 6
descriptionplural support added . . . . .	70	\glsnavhyperlink: changed 'edef to 'protected@edef . . . . . 244
symbolplural support added . . . . .	70	\glsnavhypertarget: added write to aux file . . . . . 244
\Glsentrydescplural: New . . . . .	134	\glsnavigation: changed to only use labels for groups that are present . . . . . 245
\Glsentrydescplural: New . . . . .	133	1.15 \gls@link: added \glslabel . . . . . 94
\Glsentrysymbolplural: New . . . . .	135	General: Added \glssettoctitle . . . . . 31
\Glsentrysymbolplural: New . . . . .	134	\gls@defglossaryentry: check for \@glo@first in description . . . . . 74
\SetDescriptionFootnoteAcronymStyle: Added \protect before \footnote and \glslink . . . . .	214	check for \@glo@text in symbol . . . . . 75
\SetFootnoteAcronymStyle: Added \protect before \footnote and \glslink . . . . .	220	\gls@hypergroup: new . . . . . 245
symbolplural: new . . . . .	59	\glsnavhypertarget: added check if rerun required . . . . . 244
1.13	General: Add Polish support . . . . .	349, 352
fixed bug that ignored 3rd parameter . . . . .	112–120	\glssettoctitle: new . . . . . 29
\ACRfullpl: new . . . . .	196	\printglossary: changed the way the TOC title is set . . . . . 164
\Acrfullpl: new . . . . .	196	1.16 \@GLS@: Test glossary type is \acronymtype in addition to checking if footnote option has been used . . . . . 108
\acrfullpl: new . . . . .	195	\@GLSpl: Test glossary type is \acronymtype in addition to checking if footnote option has been used . . . . . 110
\acrpluralsuffix: New . . . . .	193	\@Gls@: Test glossary type is \acronymtype in addition to checking if footnote option has been used . . . . . 107
\gls@defglossaryentry: Changed default first value . . . . .	71	
Changed default firstplural value . . . . .	71	

\@Glspl@: Test glossary type is \acronymtype in addition to checking if footnote option has been used .....	110	\glshyperlink: new .....	139
\@gls@: Test glossary type is \acronymtype in addition to checking if footnote option has been used .....	106	\glshypernumber: modified to allow material to be attached to location .....	190
\@glsdisp: Test glossary type is \acronymtype in addition to checking if footnote option has been used .....	111	\glsnavhyperlink: replaced 'hy- perlink to '@glslink .....	244
\@glspl@: Test glossary type is \acronymtype in addition to checking if footnote option has been used .....	109	\glsnavhypertarget: replaced 'hypertarget to '@glstarget .	244
\@glstarget: raised the hyper- target so the target text doesn't scroll off the top of the page	105	\glssee: new .....	162
\gls@defglossaryentry: Changed def to let .....	71	\glsseeformat: new .....	162
1.17		\glsSetSuffixF: new .....	34
\@do@wrglossary: new .....	159	\glsSetSuffixFF: new .....	34
\@do@seeglossary: new .....	161	\ifglsxindy: new .....	23
\@glo@storeentry: new .....	77	\istfilename: added xindy sup- port .....	33
\@glossary: changed defini- tion to use \index instead of \@index .....	157	\newglossarystyle: made \newglossarystyle long .	189
\@glsdefaultplural: new ....	62	\nopostdesc: new .....	32
\@glsdefaultsort: new .....	62	nonumberlist: new .....	60
\@glshypernumber: new .....	190	\printglossary: added check to determine if \printglossary is already defined .....	164
\@glsnoname: new .....	61	added print language to aux file	164
\@glsnonextpages: new .....	180	order: order package option added .....	23
\@wrglossary: modified to allow for xindy support .....	157	\writeist: added xindy support	142
General: added Brazilian dictio- nary .....	353	1.18	
Added Brazilian support ....	349	\@gls@loadlist: new .....	8
added xindy support .....	23	\@gls@loadlong: new .....	8
parent: new .....	60	\@gls@loadsuper: new .....	8
see: new .....	60	\@gls@loadtree: new .....	8
\gls@defglossaryentry: added nonumberlist key .....	71	\gls@defglossaryentry: Changed default value of sort to \@glsdefaultsort .....	71
added parent key .....	71	moved sort sanitization to \newglossaryentry .....	75
added see key .....	71	\glstarget: new .....	183
Stored main part of entry format when entry is defined .....	75	\oldacronym: new .....	192
\gls@suffixF: new .....	34	nolist: new .....	8
\gls@suffixFF: new .....	34	nolong: new .....	8
		sort: moved sanitization to \newglossaryentry .....	58
		nostyles: new .....	8
		nosuper: new .....	8
		notree: new .....	8
		1.19	
		\glsclearpage: new .....	38
		\glsdisp: new .....	111

		2.03	
\SetDescriptionAcronymStyle:			\@GLS@: Added check for hyper-
changed \acronymfont to	use \textsmaller instead of		first ..... 108
\smaller .....	218	\@GLSpl: Added check for hyper-	
\SetDescriptionFootnoteAcronymStyle:		first .....	110
changed \acronymfont to	use \textsmaller instead of	\@GLs@: Added check for hyper-	
\smaller .....	214	first .....	107
\SetFootnoteAcronymStyle:		\@GLspl@: Added check for hyper-	
changed \acronymfont to	use \textsmaller instead of	first .....	110
\smaller .....	221	\@gls@: Added check for hyper-	
\SetSmallAcronymStyle:		first .....	106
changed \acronymfont to	use \textsmaller instead of	\@gls@@link: new .....	93
\smaller .....	224	\@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 . 252,267	fied as a list of acronyms ...	107
	\glslistdottedwidth: changed	\@GLspl@: Changed test to check	
	\linewidth to \hsize ....	if glossary type has been iden-	
	251	tified as a list of acronyms ..	110
	\gls pagelistwidth: changed	\@glossaryentryfield: new ..	76
	\linewidth to \hsize . 252,267	\@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 .....	\@glsacronymlists: new .....	14
	146	\@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-	
	166	tified as a list of acronyms ...	109
	General: Changed Brazil to Brazil-		
	ian .....		
	353		
	false will prevent automatic		
	loading of translator package		
	28		
	\glossarysection: changed		
	\@mkboth to \glossarymark		
	36		
	\gls glossarymark: New .....		
	36		



\@newglossaryentryposthook:	Removed spurious brace. Patch provided by Sergiu Dotenco	111
new .....	76	
\@newglossaryentryprehook:	\writeist: Added \string before opening and closing braces. Patch provided by Segiu Dotenco	147
new .....	76	
acronymlists:new .....	15	
\DeclareAcronymList:new ...	14	
\DefineAcronymSynonyms:new	209	2.06
\gls@defglossaryentry: added		
user1-6 keys .....	71	
\glsadd: fixed bug that ignored counter .....	140	
\Glsentryuseri:new .....	136	
\Glsentryuseri:new .....	135	2.07
\Glsentryuserii:new .....	136	
\Glsentryuserii:new .....	136	
\Glsentryuseriii:new .....	136	
\Glsentryuseriii:new .....	136	
\Glsentryuseriv:new .....	136	3.0
\Glsentryuseriv:new .....	136	
\Glsentryuserv:new .....	136	
\Glsentryuserv:new .....	136	
\Glsentryuservi:new .....	137	
\Glsentryuservi:new .....	137	
\ns@newglossary: added check to determine if \gls@<type>@display and \gls@<type>@displayfirst have been defined. ....	55	
\SetAcronymLists:new .....	15	
\SetDefaultAcronymDisplayStyle:new .....	210	
\SetDefaultAcronymStyle:new .....	211	
\SetDescriptionAcronymDisplayStyle:new .....	216	
\SetDescriptionDUAAcronymDisplayStyle:new .....	215	
\SetDescriptionFootnoteAcronymDisplayStyle:new .....	212	
\SetDUADisplayStyle:new ..	224	
\SetFootnoteAcronymDisplayStyle:new .....	219	
\SetSmallAcronymDisplayStyle:new .....	221	
2.05		
\@glsdisp: Added closing brace. Patch provided by Sergiu Dotenco .....	111	
\altnewglossary:new .....	56	
\CustomAcronymFields:new ..	226	
\CustomNewAcronymDef:new ..	227	
\SetCustomDisplayStyle:new	226	
\SetCustomStyle:new .....	227	
General: glssadd format key stored in \@glsnumberformat (was mistakenly stored in \@glo@format) .....	140	
\@@do@wrglossary: added check for hyper location prefix ...	159	
modified to use new format ..	159	
\@@glossarysec: replaced \@ifundefined with \ifcsundef .....	5	
\@do@seeglossary: Sanitize and escape cross-referencing information .....	161	
\@gls@counterwithin:new ....	9	
\@gls@ifinlist:new .....	39	
\@gls@link: added \@gls@saveentrycounter .....	95	
added \@gls@setsort .....	95	
\@gls@saveentrycounter:new	95	
\@gls@setsort@def:new ...	11	
\@gls@setsort@standard:new .....	10	
\@gls@setsort@use:new ...	11	
\@gls@xdy@locationlist:new	42	
\@gls@link: replaced \@ifundefined with \ifcsundef .....	104	
\@glsnextpages:new .....	180	
\@makeglossary: Added check for savewrites .....	148	
\@print@glossary: replaced \@ifundefined with \ifcsundef .....	166	
\@printglossary: added \currentglossary .....	165	
added \@glsnextpages .....	165	

make toctitle default to title ..	165	\GlsAddXdyCounters:new ....	39
\@set@glo@numformat: added		\glentrycounterlabel:new	182
4th argument .....	97	\glentryitem:new .....	182
\@wrglossary: modified to take		\glentrylong:new .....	137
into account savewrites ....	157	\glentrylong:new .....	137
\@xdyattributelist:new ....	39	\glentrylongpl:new .....	137
General: added prefix to hyperlink		\glentrylongpl:new .....	137
.....	191	\glentryshort:new .....	137
etoolbox now loaded .....	4	\glentryshort:new .....	137
replaced \@ifundefined with		\glentryshortpl:new .....	137
\ifcsundef .....	28, 91, 177	\glentryshortpl:new .....	137
\acrfootnote:new .....	212	\glsggetgrouptitle: replaced	
\ACRfull: added starred version	195	\@ifundefined with	
\Acrfull: added starred version	195	\ifcsundef .....	187
\acrfull: added starred version	194	\glsglossarymark: replaced	
\ACRfullpl: added starred ver-		\@ifundefined with	
sion .....	196	\ifcsundef .....	36
\Acrfullpl: added starred ver-		\glshyperlink: changed de-	
sion .....	196	fault from \glentryname to	
\acrfullpl: added starred ver-		\glentrytext .....	139
sion .....	195	\glshypernumber: replaced	
\acrlinkfootnote:new ....	212	\@ifundefined with	
\acrnolinkfootnote:new ...	212	\ifcsundef .....	190
\addglossarytocaptions: re-		\glnumberformat: replaced	
placed \@ifundefined with		\@ifundefined with	
\ifcsundef .....	30	\ifcsundef .....	34
savewrites:new .....	25	\glhrefentry:new .....	182
see: added \@glo@seeautonumberlist		\glresetsubentrycounter:	
.....	60	new .....	181
seeautonumberlist:new .....	7	\glseeitem: hyperlink uses	
\glossarysection: replaced		\glseeitemformat instead	
\@ifundefined with		of \glentryname .....	163
\ifcsundef .....	36	\glseeitemformat:new ....	163
\glossarystyle: replaced		\glssortnumberfmt:new ....	10
\@ifundefined with		\glssstepentry:new .....	181
\ifcsundef .....	188	\glssstepsubentry:new ....	181
\glscodepage: replaced		\glssubentrycounterlabel:	
\@ifundefined with		new .....	182
\ifcsundef .....	24	\glssubentryitem:new ....	182
\glscdefglossaryentry: added		theglossary: replaced \@ifundefined	
\@glscdefsort .....	75	with \ifcsundef .....	183
added short and long keys ....	71	short:new .....	61
replaced \@ifundefined with		shortplural:new .....	61
\ifcsundef .....	72	\ifglossaryexists: re-	
\glscdoclearpage: replaced		placed \@ifundefined with	
\@ifundefined with		\ifcsundef .....	49
\ifcsundef .....	38	\ifglentryexists: re-	
\glscadd: added \@glscsaveentrycounter		placed \@ifundefined with	
.....	140	\ifcsundef .....	49

\istfile: deprecated .....	156	\showglossaryin: new .....	233
glossaryentry: new .....	180	\showglossaryout: new .....	233
glossarysubentry: new .....	181	\showglossarytitle: new ...	233
\newglossaryentry: replaced		\showglosymbol: new .....	231
\DeclareRobustCommand		\showglosymbolplural: new .	231
with \newrobustcmd .....	64	\showglotext: new .....	229
\newglossarystyle: re-		\showglotype: new .....	229
placed \@ifundefined with		\showglouserii: new .....	230
\ifcsundef .....	189	\showglouseriii: new .....	230
\ns@newglossary: added		\showglouseriv: new .....	230
\@gls@defsortcount .....	55	\showglouservi: new .....	230
replaced \@ifundefined with		subentrycounter: new .....	9
\ifcsundef .....	55	\writeist: added xindy-only	
entrycounter: new .....	9	macro definitions to glossary	
entrycounterwithin: new .....	9	open tag .....	144
\oldacronym: replaced \@ifundefined		modified to support new for-	
with \ifcsundef .....	192	mat .....	142
compatible-2.07: compatible-			
2.07 option added .....	25		
long: new .....	61	3.01	
longplural: new .....	61	\@gls@writefiles: added check	
nonumberlist: now boolean ...	60	for empty glossaries .....	156
sort: new .....	10	General: made robust .....	107
counter: replaced \@ifundefined		\ACRfull: made robust .....	195
with \ifcsundef .....	59	\Acrfull: made robust .....	194
\printglossary: replaced		\acrfull: made robust .....	194
\@ifundefined with		\acrfullformat: removed	
\ifcsundef .....	164	\acronymfont as it should al-	
\SetDescriptionFootnoteAcronymDisplayStyle: be set in the second ar-		gument. ....	194
expanded options link op-		\ACRfullpl: made robust ....	196
tions .....	212	\Acrfullpl: made robust ....	196
\setentrycounter: added op-		\acrfullpl: made robust ....	195
tional argument .....	188	\ACRlong: made robust .....	130
\showacronymlists: new ....	232	\Acrlong: made robust .....	129
\showglocounter: new .....	229	\acrlong: made robust .....	128
\showglodesc: new .....	231	\ACRlongpl: made robust ....	132
\showglodescplural: new ...	231	\Acrlongpl: made robust ....	131
\showglofirst: new .....	229	\acrlongpl: made robust ....	130
\showglofirstpl: new .....	229	\ACRshort: made robust .....	126
\showgloflag: new .....	232	\Acrshort: made robust .....	126
\showgloindex: new .....	232	\acrshort: made robust .....	125
\showglolevel: new .....	228	\ACRshortpl: made robust ....	128
\showgloname: new .....	231	\Acrshortpl: made robust ....	127
\showgloparent: new .....	228	\acrshortpl: made robust ....	127
\showgloplural: new .....	229	\Gls: made robust .....	107
\showglosort: new .....	231	\glsadd: made robust .....	140
\showglossaries: new .....	233	\glsaddall: made robust ....	140
\showglossarycounter: new .	233	\GLSdesc: made robust .....	117
\showglossaryentries: new .	233		

\Glsdesc: made robust .....	116	3.02	
\glsdesc: made robust .....	116		\@do@wrglossary: changed
\GLSdescplural: made robust .	118		\@glslocref to \theglsentrycounter
\Glsdescplural: made robust .	117		..... 160
\glsdescplural: made robust .	117		\@do@wrglossary: changed
\glsfirst: made robust .....	112		\@do@wr@glossary to test for
\GLSfirstplural: made robust	115		indexonlyfirst option; put old
\Glsfirstplural: made robust	115		\@do@wr@glossary code into
\glsfirstplural: made robust	114		\@do@wrglossary .....
\glslink: made robust .....	93		158
\GLSname: made robust .....	116		\@gls@missingnumberlist:
\Glsname: made robust .....	115		new .....
\glsname: made robust .....	115		62
\GLSpl: made robust .....	110		\@glswritefiles: added check
\Glspl: made robust .....	109		for existence of token in case
\glspl: made robust .....	108		\makeglossaries has been
\GLSplural: made robust ....	114		omitted .....
\GLSsymbol: made robust ....	118		156
\Glsymbol: made robust ....	118		\@printglossary: add a way to
\GLSsymbolplural: made robust			fetch current entry label ...
.....	119		166
\Glsymbolplural: made robust			\@wrglossary: added check for
.....	119		glossary file defined .....
\glsymbolplural: made robust			158
.....	119		General: added check for polyglos-
\Glstext: made robust .....	112		sia .....
\glstext: made robust .....	112		28
\GLSuseri: made robust .....	120		reversed order of package check
\Glsuseri: made robust .....	120		32
\glsuseri: made robust .....	120		savenumberlist: new .....
\GLSuserii: made robust ....	121		7
\Glsuserii: made robust ....	121		ucmark: new .....
\glsuserii: made robust ....	121		9
\GLSuseriii: made robust ....	122		\@gls@defglossaryentry: added
\Glsuseriii: made robust ....	122		numberlist element .....
\glsuseriii: made robust ....	121		74
\GLSuseriv: made robust ....	123		\@gls@save@numberlist: new .
\Glsuseriv: made robust ....	123		163
\glsuseriv: made robust ....	122		\@glsdisplaynumberlist: new
\GLSuseriv: made robust .....	124		138
\Glsuseriv: made robust .....	123		\@glsentrycounter: set default
\GLSuseriv: made robust .....	124		value .....
\glsuseriv: made robust ....	122		95
\GLSuseriv: made robust .....	124		\@glsentryfull: fixed bug (re-
\Glsuseriv: made robust .....	123		placed \@glsentryshortpl
\GLSuseriv: made robust .....	124		with \@glsentryshort) ....
\glsuseriv: made robust ....	122		138
\GLSuseriv: made robust .....	124		\@glsentryfullpl: fixed bug (re-
\Glsuseriv: made robust .....	123		placed \@glsentryshort with
\GLSuseriv: made robust .....	124		\@glsentryshortpl) .....
\glsuseriv: made robust ....	122		138
\GLSuseriv: made robust .....	124		\@glsentrynumberlist: new ..
\Glsuseriv: made robust .....	123		138
\GLSuseriv: made robust .....	124		\@glsmoveentry: new .....
\glsuseriv: made robust ....	122		76
\GLSuseriv: made robust .....	124		\@glsnumlistlastsep: new ...
\Glsuseriv: made robust .....	123		139
\GLSuseriv: made robust .....	124		\@glsnumlistsep: new .....
\glsuseriv: made robust ....	122		139
\GLSuseriv: made robust .....	124		\@glsresetsubentrycounter:
\Glsuseriv: made robust .....	123		new .....
\GLSuseriv: made robust .....	124		181
\glsuseriv: made robust ....	122		\@ifglshaschildren: new .....
\GLSuseriv: made robust .....	124		50
\Glsuseriv: made robust .....	123		\@ifglshasparent: new .....
\GLSuseriv: made robust .....	124		51
\glsuseriv: made robust ....	122		\@makeglossaries: added list
\GLSuseriv: made robust .....	124		parser .....
\Glsuseriv: made robust .....	123		151
\GLSuseriv: made robust .....	124		indexonlyfirst: new .....
\glsuseriv: made robust ....	122		22
\GLSuseriv: made robust .....	124		\@renewglossarystyle: new ..
\Glsuseriv: made robust .....	123		189

\showglossaryentries: fixed misspelt command .....	233	super3col: added check for glsnogroupskip .....	270
\SmallNewAcronymDef: fixed broken short and long plural	222	super4col: added check for glsnogroupskip .....	271
3.03		superragged: added check for glsnogroupskip .....	275
\@gls@sanitizesort:new ....	18	superragged3col: added check for glsnogroupskip .....	277
\@gls@setupsort@standard: used \@gls@sanitizesort .	10	3.04	
\@printglossary: allow title to override default toctitle ....	165	\@do@wrglossary: changed \theglsentrycounter back to \@glslocref .....	160
General: allow title to set toctitle	177	\@do@wrglossary: modified to compensate for possible incorrect page number .....	159
\glsinlinedescformat:new .	248	\@gls@escbsdq: unsanitize \gls@numberpage, \gls@alphpage, \gls@Alphpage and \gls@romanpage .....	98
\glsinlineemptydescformat: new .....	248	\@print@glossary: Moved aux write to end of document to prevent unwanted whatsit occurring here. ....	166
\glsinlinenameformat:new .	248	General: Added check for doc package .....	4
\glsinlinepostchild:new ..	248	added datatool-base as a required package .....	4
\glsinlinesubdescformat: new .....	248	added local key .....	92
\glsinlinesubnameformat: new .....	248	\gls@Alphpage:new .....	158
\glspostinline: replaced “.” with \glspostdescription	248	\gls@alphpage:new .....	158
altlongragged4col: added check for glsnogroupskip ..	262	\gls@disablepagerefexpansion: new .....	158
altsuperragged4col: added check for glsnogroupskip ..	278	\gls@numberpage:new .....	158
alttree: added check for glsnogroupskip .....	286	\gls@protected@pagefmts: new .....	158
index: added check for glsnogroupskip .....	281	\gls@romanpage:new .....	159
nogroupskip:new .....	9	\glsdefmain: added check for doc package .....	12
long: added check for glsnogroupskip .....	252	\glsorg@endtheglossary:new .	5
long3col: added check for glsnogroupskip .....	254	\glsorg@glossary:new .....	4
long4col: added check for glsnogroupskip .....	255	\glsorg@theglossary:new ....	5
longragged: added check for glsnogroupskip .....	259	\glsorg@wrglossary:new ....	4
longragged3col: added check for glsnogroupskip .....	260	altlist: replaced \newline with paragraph break .....	250
nopostdot:new .....	9	\PrintChanges:new .....	5
tree: added check for glsnogroupskip .....	282	3.05	
treenoname: added check for glsnogroupskip .....	284	\@do@wrglossary: add Roman case. Fixed bugs in the else statements .....	159
super: added check for glsnogroupskip .....	268		

\@gls@link: added check for “no-hypertypes” .....	94	General: added nogroupskip key to \printglossary .....	178
\@gls@nohyperlist: new .....	16	removed definition of \@glossaryentryfield ..	327
mcolalttree: replaced ‘2’ with \glsmcols .....	266	removed definition of \@glossarysubentryfield	327
mcolindex: replaced ‘2’ with \glsmcols .....	264	\compatibleglossentry: new	183
mcoltree: replaced ‘2’ with \glsmcols .....	264	\compatiblesubglossentry: new .....	185
mcoltreenoname: replaced ‘2’ with \glsmcols .....	265	\glossaryentryfield: deprecated .....	185
\gls@protected@pagefmts: added Roman to list .....	158	\Glossentrydesc: new .....	184
\gls@Romanpage: new .....	159	\glossentrydesc: new .....	184
\GlsDeclareNoHyperList: new	16	\Glossentryname: new .....	184
\glsgetgrouplabel: fixed bug (typo in \equal) .....	188	\glossentryname: new .....	184
\nopostdesc: made robust .....	32	\Glossentrysymbol: new ....	184
nohypertypes: new .....	16	\glossentrysymbol: new ....	184
3.06		\gls@assign@desc@field: new	17
\@xdy@main@language: Changed back to using \language	24	\gls@assign@desc@plural@field: new .....	17
\findrootlanguage: Obsoleted	46	\gls@assign@field: new .....	64
3.07		\gls@ifnotmeasuring: new ...	78
\@gls@link: fixed bug that failed to find entry in list .....	94	\glsaddallunused: new .....	140
\glossarypreamble: modified to work with \setglossarypreamble .....	35	\glsexpandfields: new .....	64
\gls@doclearpage: added check for openright .....	38	\glsnoexpandfields: new ....	64
\glspostdescription: Added spacefactor code .....	8	\glssee: made robust .....	162
\GlsSetXdyCodePage: Added check for fontspec .....	47	\glsseeformat: made robust ..	162
\SetDescriptionAcronymDisplayStyle: now using \glsdoparenifnotempty .....	216	\glsseeitem: made robust ....	163
\setglossarypreamble: new ..	35	\glsseelist: made robust ....	162
3.08a		\ifglstdescsuppressed: new ..	51
\@gls@storeentry: no longer need to check for special characters in any of the fields other than sort .....	77	\ifglshasdesc: new .....	51
updated for \glossentry ....	77	\ifglshassymbol: new .....	51
\@glossaryentryfield: switched to \glossentry .....	76	list: updated list style to use \glossentry and \subglossentry .....	249
\@glossarysubentryfield: switched to \subglossentry	76	listdotted: updated listdotted style to use \glossentry and \subglossentry .....	251
		altlist: updated altlist style to use \glossentry and \subglossentry .....	250
		altlongragged4col: updated to use \glossentry and \subglossentry .....	261
		alttree: updated to use \glossentry and \subglossentry .....	285
		index: added paragraph break at end of environment .....	280

updated to use \glossentry and \subglossentry .....	280	\Glsentryuser: made robust .	136
inline: updated inline style to use \glossentry and \subglossentry .....	246	\Glsentryuseri: made robust	137
long: updated to use \glossentry and \subglossentry .....	252	\glstextup: new .....	193
longragged: updated to use \glossentry and \subglossentry .....	258	\if@gl@docloaded: Add a fix for \RecordChanges .....	4
longragged3col: updated to use \glossentry and \subglossentry .....	260	\ifglshassymbol: changed test to check for \@gl@default@symbol .....	51
tree: updated to use \glossentry and \subglossentry .....	282	3.10a	
\setglossarystyle: new ....	188	\@gl@keymap: new .....	66
\setglossentrycompatibility: new .....	185	\@gl@provide@newglossary: new .....	53
superragged: updated to use \glossentry and \subglossentry .....	275	\@gl@writedef: new .....	65
3.09a		\@gl@defaultplural: Obsolete .	62
\@gl@assign@symbolplural@field: new .....	18	\@gl@nodesc: new .....	62
\@gl@default@value: new ...	59	\@print@glossary: Added providecommand code to aux file .....	167
\Glsentrydesc: made robust ..	133	\gl@assign@type@field: new	17
\Glsentrydescplural: made ro- bust .....	134	\gl@defglossaryentry: Changed to using \@gl@default@value .....	71
\Glsentryfirst: made robust .	135	new .....	70
\Glsentryfirstplural: made robust .....	135	\glswritedefhook: new .....	69
\Glsentryfull: made robust ..	138	\makeglossaries: Added providecommand code to aux file .....	150
\Glsentryfullpl: made robust	138	\new@glossaryentry: new ....	65
\Glsentrylong: made robust ..	137	\ns@newglossary: added \@gl@provide@newglossary .....	55
\Glsentrylongpl: made robust	137	3.11a	
\Glsentryname: made robust ..	133	\@ACRlong: added \glslabel, \gl@ifplural, \glscapscase, \gl\$insert and \glscustomtext .....	326
\Glsentryplural: made robust	134	\@ACRshort: added \glslabel, \gl@ifplural, \glscapscase, \gl\$insert and \glscustomtext .....	325
\Glsentryshort: made robust .	137	\@Acrlong: added \glslabel, \gl@ifplural, \glscapscase, \gl\$insert and \glscustomtext .....	326
\Glsentryshortpl: made robust .....	137	\@Acrshort: added \glslabel, \gl@ifplural, \glscapscase, \gl\$insert and \glscustomtext .....	325
\Glsentrysymbol: made robust	134		
\Glsentrysymbolplural: made robust .....	135		
\Glsentrytext: made robust ..	134		
\Glsentryuseri: made robust .	136		
\Glsentryuserii: made robust	136		
\Glsentryuseriii: made robust .....	136		
\Glsentryuseriv: made robust	136		

<p>\@GLS@: add \glslabel,  \glsifplural,\glscapscase,  \glscustomtext and  \glsinsert ..... 108  change to using \glsentryfmt  style commands ..... 108  removed \MakeUppercase  (now moved to \glsentryfmt)  ..... 108  \@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  ..... 326  \@acrshort: added \glslabel,  \glsifplural,\glscapscase,  \glsinsert and \glscustomtext  ..... 325  \@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 245
..... 122	\ifglshasdesc: replaced
changed to just use \glentryuseriii	\ifdefempty with \ifcsemtyp
..... 122	..... 51
changed to just use \Glsentryuserii	\ifglshaslong: new ..... 51
..... 121	\ifglshasshort: new ..... 52
changed to just use \glentryuserii	\ifglshassymbol: replaced
..... 121	\ifdefempty with \ifcsemtyp
changed to just use \Glsentryuseriv	..... 51
..... 123	\ifglused: replaced \ifthenelse
changed to just use \glentryuseriv	with \ifbool ..... 49
..... 123	\longnewglossaryentry: new . 69
changed to just use \Glsentryuseri	\ns@newglossary: replaced
..... 120	\glssdisplay and \glssdisplayfirst
changed to just use \glentryuseri	with \glssentryfmt ..... 55
..... 120	compatible-3.07: cnew ..... 25
changed to just use \Glsentryuservi	\SetCustomDisplayStyle: up-
..... 125	dated to use \defglssentryfmt
changed to just use \glentryuservi	..... 226
..... 124, 125	\SetDefaultAcronymDisplayStyle:
changed to just use \Glsentryuserv	changed to use \defglssentryfmt
..... 124	..... 210
changed to just use \glssentryserv	\SetDescriptionAcronymDisplayStyle:
..... 123, 124	updated to use \defglssentryfmt
Now requires textcase ..... 3	..... 216
acronymlists: replaced	\SetDescriptionDUAAcronymDisplayStyle:
\@addtoacronymlists with	updated to use \defglssentryfmt
\DeclareAcronymList .... 15	..... 215
\defglssdisplay: obsoleted .... 90	\SetDescriptionFootnoteAcronymDisplayStyle:
\defglssdisplayfirst: obso-	updated to use \defglssentryfmt
leted ..... 90	..... 212
\defglssentryfmt: new ..... 54	\SetDUADisplayStyle: updated
\forglssentries: replaced \ifx	to use \defglssentryfmt .. 224
with \ifdefempty ..... 48	\SetFootnoteAcronymDisplayStyle:
\gls@assign@desc: new ..... 69	updated to use \defglssentryfmt
\gls@defglossaryentry: Fixed	..... 219
default counter if none sup-	\SetSmallAcronymDisplayStyle:
plied ..... 74	updated to use \defglssentryfmt
\gls@doentryfmt: new ..... 54	..... 221
\glssdisplay: obsoleted ..... 90	\setupglossaries: new ..... 27
\glssdisplayfirst: obsoleted .. 90	\showglolong: new ..... 232
\glssgenentryfmt: new ..... 85	\showgloshort: new ..... 232
\glssgetgrouptitle: Added	numbers: new ..... 26
check in case non-Latin alpha-	symbols: new ..... 25
bet in use ..... 187	

3.12a		\glsentryfullpl: changed to use \acrfullformat ..... 138
\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 ..... 273
..... 18		altsuper4colheaderborder:
\@gls@link: removed \relax .. 95		switched to \tabularnewline
\@gls@notranslatorhook: new 21		..... 273
\@gls@setupsort@standard:		long: switched to \tabularnewline
moved \@gls@santizesort		..... 252
to \glsprestandardsort .. 10		long3col: switched to \tabularnewline
General: added cs@gls@notranslatorhook		..... 254
to else clause ..... 32		long3colheader: switched to
ucmark: added check for memoir . 9		\tabularnewline ..... 254
see: added \gls@checkseeallowed		long3colheaderborder: switched
..... 60		to \tabularnewline ..... 255
\glossarysection: changed		long4col: switched to \tabularnewline
\glossarymark to \gls glossarymark		..... 255
..... 36		long4colheader: switched to
\glossarystyle: fixed bug		\tabularnewline ..... 256
caused by using \ifdef in-		longheader: switched to
stead of \ifcsdef ..... 188		\tabularnewline ..... 253
\gls@assign@desc@field:		longheaderborder: switched to
changed to use \glssetnoexpandfield		\tabularnewline ..... 253
..... 17		\SetFootnoteAcronymDisplayStyle:
\gls@assign@descplural@field:		fixed missing argument bug 219
changed to use \glssetnoexpandfieldsuper:		switched to \tabularnewline
..... 17		..... 268
\gls@assign@name@field:		super3col: switched to
changed to use \glssetnoexpandfield		\tabularnewline ..... 269
..... 17		super3colheader: switched to
\gls@assign@type@field:		\tabularnewline ..... 270
changed to use \glssetexpandfield		super4col: switched to
..... 17		\tabularnewline ..... 271
\gls@checkseeallowed: new .. 60		super4colheader: switched to
\glsaddallunused: set default to		\tabularnewline ..... 271
\@glo@types ..... 140		super4colheaderborder:
\Glsentryfull: changed to use		switched to \tabularnewline
\acrfullformat ..... 138		..... 272
\glsentryfull: changed to use		superheader: switched to
\acrfullformat ..... 138		\tabularnewline ..... 268
\Glsentryfullpl: changed to		superheaderborder: switched to
use \acrfullformat ..... 138		\tabularnewline ..... 269

3.14a		\genacrfullformat:new . . . . . 89
\@glswritefiles:	renamed	\GenericAcronymFields:new 198
\glswritefiles to \@glswritefiles		\Genplacrfullformat:new ... 90
and used “savewrites” option		\genplacrfullformat:new ... 89
to set \@glswritefiles . . . . .	156	\Glsentryfull: bug fix: added
General:new . . . . .	234	missing\acronymfont . . . . . 138
acronyms:new . . . . .	14	\Glsentryfull: bug fix: added
\gls@defglossaryentry: added		missing\acronymfont . . . . . 138
check for existence of default		\Glsentryfullpl: bug fix: added
glossary . . . . .	71	missing\acronymfont . . . . . 138
set the default for firstplural to		\Glsentryfullpl: bug fix: added
be the value of plural . . . . .	73	missing\acronymfont . . . . . 138
xindygloss:new . . . . .	24	\glsgenacfmt:new . . . . . 87
\longprovideglossaryentry:		\GlsUseAcrEntryDisplayStyle:
new . . . . .	70	new . . . . . 200
compatible-2.07: added check		\GlsUseAcrStyleDefs:new .. 200
for 2.07 before setting 3.07		short-long:new . . . . . 200
compatibility . . . . .	25	short-long-desc:new . . . . . 203
notranslate:new . . . . .	21	xindynoglsnumbers:new . . . . . 24
\provideglossaryentry:new . . . . .	64	sm-short-long:new . . . . . 202
4.0		sm-short-long-desc:new ... 203
\gls@defglossaryentry: added		\makeglossaries: made pream-
check for first key . . . . .	73	ble only . . . . . 151
4.01		index:new . . . . . 26
General: fixed non-value options		\newacronymstyle:new . . . . . 199
so that they can be passed to		long-sc-short:new . . . . . 201
document class . . . . .	7	long-sc-short-desc:new ... 202
\CustomAcronymFields:	in-	long-short:new . . . . . 200
serted missing comma . . . . .	227	long-short-desc:new . . . . . 202
4.02		long-sm-short:new . . . . . 201
\@acrfull: now using \@acrfullfmt		long-sm-short-desc:new ... 203
. . . . .	194	footnote:new . . . . . 206
\@gls@indexdef:new . . . . .	26	footnote-desc:new . . . . . 208
\@gls@numbersdef:new . . . . .	26	footnote-sc:new . . . . . 207
\@gls@symbolsdef:new . . . . .	26	footnote-sc-desc:new . . . . . 208
General: Removed \@acronymfont		footnote-sm:new . . . . . 207
. . . . .	129–132	footnote-sm-desc:new . . . . . 208
\ACRfullfmt:new . . . . .	195	\setacronymstyle:new . . . . . 199
\Acrfullfmt:new . . . . .	195	\SetDescriptionAcronymDisplayStyle:
\acrfullfmt:new . . . . .	194	Moved check for empty cus-
\ACRfullplfmt:new . . . . .	196	tom text to prevent unwanted
\Acrfullplfmt:new . . . . .	196	parenthetical material . . . . . 216
\acrfullplfmt:new . . . . .	196	\SetDescriptionFootnoteAcronymDisplayStyle:
\acronymentry:new . . . . .	198	Moved check for empty cus-
sanitize: fixed bug that caused		tom text to prevent unwanted
an error here . . . . .	21	parenthetical material . . . . . 212
sc-short-long:new . . . . .	202	\SetFootnoteAcronymDisplayStyle:
sc-short-long-desc:new ...	203	Moved check for empty cus-
\Genacrfullformat:new . . . . .	89	

tom text to prevent unwanted parenthetical material . . . . .	219	\@glsdisp: removed \glslabel (defined in \@gls@link) . . .	111
\SetGenericNewAcronym: new	197	\@glspl@: removed \glslabel (defined in \@gls@link) . . .	109
\SetSmallAcronymDisplayStyle: Moved check for empty cus- tom text to prevent unwanted parenthetical material . . . . .	221	\@printglossary: added \glsdetoklabel . . . . .	166
dua: new . . . . .	204	General: changed default to \@empty instead of \relax . .	25
dua-desc: new . . . . .	206	removed \glslabel (defined in \@gls@link) . . . . .	125
numberedsection: added		sc-short-long-desc: redefined to use accessibility informa- tion . . . . .	331
nameref option . . . . .	6	\compatibleglossentry: added \glsdetoklabel . . . . .	307
4.03		\compatiblesubglossentry: added \glsdetoklabel . . .	308
\@do@wrglossary: added		\Genacrfullformat: redefined to use accessibility informa- tion . . . . .	324
\glsdetoklabel . . . . .	160	\genacrfullformat: redefined to use accessibility informa- tion . . . . .	324
\@ACRlong: removed \glslabel (defined in \@gls@link) . . .	326	\Genplacrfullformat: rede- fined to use accessibility in- formation . . . . .	324
\@ACRshort: removed \glslabel (defined in \@gls@link) . . .	325	\genplacrfullformat: rede- fined to use accessibility in- formation . . . . .	324
\@Acrlong: removed \glslabel (defined in \@gls@link) . . .	326	\glossentryname: added \glsdetoklabel . . . . .	184
\@Acrshort: removed \glslabel (defined in \@gls@link) . . .	325	\gls@defglossaryentry: added \glsdetoklabel . . . . .	70
\@GLS@: removed \glslabel (de- fined in \@gls@link) . . . . .	108	replaced #1 with \@glo@label	71
\@GLSpl: removed \glslabel (defined in \@gls@link) . . .	110	replaced \ifthenelse with \ifdefequal . . . . .	72
\@Gls@: removed \glslabel (de- fined in \@gls@link) . . . . .	107	\glsadd: added \glsdetoklabel . . . . .	140
\@Gls@entry@field: new . . .	133	\glsaddkey: switched to using \@gls@field@link . . . . .	68
\@Glspl@: removed \glslabel (defined in \@gls@link) . . .	109	\glsdetoklabel: new . . . . .	49
\@acrlong: removed \glslabel (defined in \@gls@link) . . .	326	\glsdisplaynumberlist: added \glsdetoklabel . . . . .	139
\@acrshort: removed \glslabel (defined in \@gls@link) . . .	325	\glsdoifexistsorwarn: new . .	50
\@gls@: removed \glslabel (de- fined in \@gls@link) . . . . .	106	\glsentryaccess: switched to using \@gls@entry@field .	312
\@gls@access@display: new .	313	\glsentrydescaccess: switched to using \@gls@entry@field	312
\@gls@entry@field: new . . .	132		
\@gls@fetchfield: new . . . . .	66		
\@gls@field@link: new . . . . .	111		
\@gls@link: added \glsdetoklabel . . . . .	94		
moved \@gls@link@opts and \@gls@link@label to \@gls@link . . . . .	94		
\@gls@writedef: added			
\glsdetoklabel . . . . .	65		

<code>\glsentrydescpluralaccess:</code> switched to using <code>\@gls@entry@field</code> ..... 312	<code>\glsstepentry:added\glsdetoklabel</code> ..... 181
<code>\glsentryfirstaccess:switched</code> to using <code>\@gls@entry@field</code> 312	<code>\glsstepsubentry: added</code> <code>\glsdetoklabel</code> ..... 181
<code>\glsentryfirstplural: added</code> <code>\glsdetoklabel</code> ..... 135	<code>\glsunset:added\glsdetoklabel</code> ..... 79
<code>\glsentrylongaccess: switched</code> to using <code>\@gls@entry@field</code> 313	short-long: commented spuri- ous EOL ..... 201
<code>\glsentrylongpluralaccess:</code> switched to using <code>\@gls@entry@field</code> ..... 313	redefined to use accessibility in- formation ..... 329
<code>\glsentrypluralaccess:</code> switched to using <code>\@gls@entry@field</code> ..... 312	short-long-desc: redefined to use accessibility information 331
<code>\glsentryshortaccess:switched</code> to using <code>\@gls@entry@field</code> 313	<code>\ifglsdessuppressed: added</code> <code>\glsdetoklabel</code> ..... 51
<code>\glsentryshortpluralaccess:</code> switched to using <code>\@gls@entry@field</code> ..... 313	fixed typo ..... 51
<code>\glsentrysymbolaccess:</code> switched to using <code>\@gls@entry@field</code> ..... 312	<code>\ifglshaschildren: added</code> <code>\glsdetoklabel</code> ..... 49
<code>\glsentrysymbolpluralaccess:</code> switched to using <code>\@gls@entry@field</code> ..... 312	<code>\ifglshasdesc:added\glsdetoklabel</code> ..... 51
<code>\glsentrytextaccess: switched</code> to using <code>\@gls@entry@field</code> 312	<code>\ifglshasfield:new</code> ..... 52
<code>\glsgenacfmt: redefined to use</code> accessibility information ... 322	<code>\ifglshaslong:added\glsdetoklabel</code> ..... 51
<code>\glsgenentryfmt: redefined to</code> use accessibility information 319	<code>\ifglshasparent: added</code> <code>\glsdetoklabel</code> ..... 51
<code>\glshyperlink:added\glsdetoklabel</code> ..... 139	<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>\ifcsempy</code> with <code>\ifdefempty</code> and replaced <code>\ifx</code> with <code>\ifdefequal</code> .... 51
<code>\glsmoveentry:added\glsdetoklabel</code> ..... 76	<code>\ifglshasused:added\glsdetoklabel</code> ..... 49
replaced <code>\ifthenelse</code> with <code>\ifdefequal</code> ..... 76	sm-short-long-desc: redefined to use accessibility informa- tion ..... 331
<code>\glsrefentry:added\glsdetoklabel</code> ..... 182	long-sc-short-desc: redefined to use accessibility informa- tion ..... 330
<code>\glsreset:added\glsdetoklabel</code> ..... 78	long-short: redefined to use ac- cessibility information .... 328
<code>\glsseelist:added\expandafter</code> commands ..... 163	long-short-desc: redefined to use accessibility information 330
	long-sm-short-desc: redefined to use accessibility informa- tion ..... 330

footnote: redefined to use accessibility information . . . . .	334	\showgloplural: added	
footnote-desc: redefined to use accessibility information . . .	336	\glsdetoklabel . . . . .	229
footnote-sc: redefined to use accessibility information . . . .	336	\showglopluralaccess: added	
footnote-sc-desc: redefined to use accessibility information	337	\glsdetoklabel . . . . .	343
footnote-sm: redefined to use accessibility information . . . .	336	\showgloshort: added \glsdetoklabel	
footnote-sm-desc: redefined to use accessibility information	337	.....	232
\renewacronymstyle: new . . .	199	\showgloshortaccess: added	
\showglocounter: added		\glsdetoklabel . . . . .	343
\glsdetoklabel . . . . .	229	\showgloshortpluralaccess: added \glsdetoklabel . . .	343
\showglodesc: added \glsdetoklabel		\showglosort: added \glsdetoklabel	
.....	231	.....	231
\showglodescaccess: added		\showglosymbol: added	
\glsdetoklabel . . . . .	343	\glsdetoklabel . . . . .	231
\showglodescplural: added		\showglosymbolaccess: added	
\glsdetoklabel . . . . .	231	\glsdetoklabel . . . . .	343
\showglodescpluralaccess: added \glsdetoklabel . . .	343	\showglosymbolplural: added	
\showglofirst: added \glsdetoklabel		\glsdetoklabel . . . . .	231
.....	229	\showglosymbolpluralaccess: added \glsdetoklabel . . .	343
\showglofirstaccess: added		\showglotext: added \glsdetoklabel	
\glsdetoklabel . . . . .	343	.....	229
\showglofirstpl: added		\showglotextaccess: added	
\glsdetoklabel . . . . .	229	\glsdetoklabel . . . . .	343
\showglofirstpluralaccess: added \glsdetoklabel . . .	343	\showglotype: added \glsdetoklabel	
\showgloflag: added \glsdetoklabel		.....	229
.....	232	\showglouserii: added	
\showgloindex: added \glsdetoklabel		\glsdetoklabel . . . . .	230
.....	232	\showglouseriii: added	
\showglolevel: added \glsdetoklabel		\glsdetoklabel . . . . .	230
.....	228	\showglouseriv: added	
\showglolong: added \glsdetoklabel		\glsdetoklabel . . . . .	230
.....	232	\showglouserv: added \glsdetoklabel	
\showglolongaccess: added		.....	230
\glsdetoklabel . . . . .	344	\showglouservi: added	
\showglolongpluralaccess: added \glsdetoklabel . . .	344	\glsdetoklabel . . . . .	230
\showglongname: added \glsdetoklabel		dua: fixed bug in \acrfullfmt .	205
.....	231	fixed bug in \Acrfullplfmt .	205
\showglongnameaccess: added		fixed bug in \acrfullplfmt .	205
\glsdetoklabel . . . . .	342	redefined to use accessibility information . . . . .	331
\showgloparent: added		dua-desc: commented spurious EOL . . . . .	206
\glsdetoklabel . . . . .	228	redefined to use accessibility information . . . . .	334

4.04		
\@gls@noidx@nosanitizesort:		\def as is may or may not be
new .....	18	defined .....
\@gls@noidx@sanitizesort:		307
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 ....	168	defined .....
\@glo@do@sortentries:new ..	169	308
\@glo@grabfirst:new .....	174	\defglsdisplayfirst: fixed un-
\@glo@sortedinsert:new ...	169	wanted space .....
\@glo@sortentries:new ....	168	91
\@glo@sorthandler@case:new	170	\glo@grabfirst:new .....
\@glo@sorthandler@letter:		173
new .....	170	\gls@defglossaryentry: re-
\@glo@sorthandler@nocase:		placed \ifx with \ifdefvoid
new .....	170	75
\@glo@sorthandler@word:new	170	\glsnoidxdisplayloc:new ..
\@glo@sortmacro@case:new ..	171	176
\@glo@sortmacro@def:new ..	172	\glsnoidxdisplaylocclisthandler:
\@glo@sortmacro@def@do:new	172	new .....
\@glo@sortmacro@letter:new	171	176
\@glo@sortmacro@nocase:new	172	\glsnoidxloclist:new .....
\@glo@sortmacro@standard:		175
new .....	171	\glsnoidxlocclisthandler:
\@glo@sortmacro@use:new ..	173	new .....
\@glo@sortmacro@word:new ..	170	176
\@gls@getcounterprefix:		\glsnoidxstripaccents:new ..
added warning if no prefix can		19
be formed .....	161	alttree: moved hangindent and
\@gls@getothergrouptitle:		parindent assignments out-
new .....	187	side level test .....
\@gls@noidx@do:new .....	174	285
\@gls@noref@warn:new ....	155	\makeglossaries: Moved def-
\@gls@reference:new .....	176	inition of \glswrite to
\@gls@warnonglossdefined:		\makeglossaries .....
new .....	16	150
\@gls@warnontheGLOSSdefined:		\makenoidxglossaries:new ..
new .....	16	152
\@no@makeglossaries:new ..	155	\printglossary: changed to use
\@print@glossary:new ....	166	new \@printglossary ...
\@print@noidx@glossary:new	173	164
\@printgloss@setsort:new ..	164	\printnoidxglossaries:new
\@printglossary:new .....	165	164
General: added sort key to print-		\@printnoidxglossary:new ..
gloss group .....	179	164
\compatibleglossentry:		\showgloclist:new .....
changed \newcommand to		232
		\warn@noprintglossary: Acti-
		vate warning in \makeglossaries
		.....
		164
		\writeist: checked for definition
		of \glswrite .....
		142, 146
	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 .....	174	\@glsdisp: moved \@glsifhyper .....	111
\Acrfullplfmt: fixed no case change bug .....	196	moved check for first use to \@gls@link .....	111
\glsletentryfield: new ....	132	\@glspl@: moved \@glsifhyper .....	109
4.08		moved check for first use to \@gls@link .....	109
\@ACRlong: added \@do@gls@link@checkfirsthyper .....	326	\@ignored@glossaries: new ..	57
\@ACRshort: added \@do@gls@link@checkfirsthyper .....	325	General: added entrycounter option to printgloss family ..	178
\@Acrlong: added \@do@gls@link@checkfirsthyper .....	326	added \@nopostdot option to printgloss family .....	178
\@Acrshort: added \@do@gls@link@checkfirsthyper .....	325	added \@subentrycounter option to printgloss family .....	179
\@GLS@: moved \@glsifhyper ..	108	explicitly initialise hyper key ..	92
moved check for first use to \@gls@link .....	108	moved \@glsifhyper ...	126–132
\@GLSpl: moved \@glsifhyper ..	110	removed \@sACRlongpl ....	132
moved check for first use to \@gls@link .....	110	removed \@sAcrlongpl ....	131
\@GLS@: moved \@glsifhyper ..	107	removed \@sacrlongpl ....	130
moved check for first use to \@gls@link .....	107	removed \@sACRlong .....	130
\@GLspl@: moved \@glsifhyper ..	109	removed \@sAcrlong .....	129
moved check for first use to \@gls@link .....	109	removed \@sacrlong .....	129
\@acrlong: added \@do@gls@link@checkfirsthyper .....	326	removed \@sACRshortpl ...	128
\@acrshort: added \@do@gls@link@checkfirsthyper .....	325	removed \@sAcrshortpl ...	127
\@closegls: new .....	149	removed \@sacrshortpl ...	127
\@gls@: moved \@glsifhyper ..	106	removed \@sACRshort .....	126
moved check for first use to \@gls@link .....	106	removed \@sAcrshort .....	126
\@gls@doautomake: new .....	25	removed \@sacrshort .....	125
\@gls@field@link: added as-signment of \@do@gls@link@checkfirsthyper .....	111	removed \@sgls@link .....	93
		removed \@sGLSdescplural ..	118
		removed \@sGlsdescplural ..	117
		removed \@sglsdescplural ..	117
		removed \@sGLSdesc .....	117
		removed \@sGlsdesc .....	116
		removed \@sglsdesc .....	116
		removed \@sglsdisp .....	111
		removed \@sGLSfirstplural ..	115
		removed \@sGlsfirstplural ..	115
		removed \@sglsfirstplural ..	114
		removed \@sGLSfirst .....	113
		removed \@sGlsfirst .....	113
		removed \@sglsfirst .....	113



removed \@sGLSname .....	116	removed spglspl .....	237
removed \@sGlsname .....	116	\ACRfull: removed \s@ACRfull	195
removed \@sglsname .....	115	switched to using \@gls@hyp@opt	
removed \@sGLSplural ....	114	.....	195
removed \@sGlsplural ....	114	\Acrfull: removed \s@Acrfull	195
removed \@sglsplural ....	113	switched to using \@gls@hyp@opt	
removed \@sGLSpl .....	110	.....	194
removed \@sGlspl .....	109	\acrfull: removed \s@acrfull	194
removed \@sglspl .....	108	switched to using \@gls@hyp@opt	
removed \@sGLSsymbolplural		.....	194
.....	119	\ACRfullpl: removed \s@ACRfullpl	
removed \@sGLssymbolplural		.....	196
.....	119	switched to using \@gls@hyp@opt	
removed \@sglssymbolplural		.....	196
.....	119	\Acrfullpl: removed \s@Acrfullpl	
removed \@sGLSsymbol ....	119	.....	196
removed \@sGLssymbol ....	118	switched to using \@gls@hyp@opt	
removed \@sglssymbol ....	118	.....	196
removed \@sGLStext .....	112	\acrfullpl: removed \s@acrfullpl	
removed \@sGlstext .....	112	.....	195
removed \@sglstext .....	112	switched to using \@gls@hyp@opt	
removed \@sGLSuseriii ...	122	.....	195
removed \@sGlsuseriii ...	122	\ACRlong: switched to using	
removed \@sGLSuserii ....	121	\@gls@hyp@opt .....	130
removed \@sGlsuserii ....	121	\Acrlong: switched to using	
removed \@sglsuserii ....	121	\@gls@hyp@opt .....	129
removed \@sGLSuseriv ....	123	\acrlong: switched to using	
removed \@sGlsuseriv ....	123	\@gls@hyp@opt .....	128
removed \@sglsuseriv ....	122	\ACRlongpl: switched to using	
removed \@sGLSuseri .....	120	\@gls@hyp@opt .....	132
removed \@sGlsuseri .....	120	\Acrlongpl: switched to using	
removed \@sglsuseri .....	120	\@gls@hyp@opt .....	131
removed \@sGLSuservi ....	125	\acrlongpl: switched to using	
removed \@sGlsuservi ....	125	\@gls@hyp@opt .....	130
removed \@sglsuservi ....	124	\ACRshort: switched to using	
removed \@sGLSuserv .....	124	\@gls@hyp@opt .....	126
removed \@sGlsuserv .....	124	\Acrshort: switched to using	
removed \@sglsuserv .....	123	\@gls@hyp@opt .....	126
removed \@sGLS .....	108	\acrshort: switched to using	
removed \@sGls .....	107	\@gls@hyp@opt .....	125
removed \@sgls .....	106	\ACRshortpl: switched to using	
removed \@thirdofthree (de-		\@gls@hyp@opt .....	128
finied in kernel) .....	105	\Acrshortpl: switched to using	
removed sPGLS .....	239	\@gls@hyp@opt .....	127
removed sPgls .....	238	\acrshortpl: switched to using	
removed spgls .....	236	\@gls@hyp@opt .....	127
removed sPGLSpl .....	240	\forallacronyms: new .....	48
removed sPglspl .....	238	\GLS: switched to using	
		\@gls@hyp@opt .....	107

\Gls: switched to using \@gls@hyp@opt ..... 107	\glslink: switched to using \@gls@hyp@opt ..... 93
\gls: switched to using \@gls@hyp@opt ..... 106	\glslinkcheckfirsthyperhook: new ..... 94
\gls@defglossaryentry: added check for ignored glossary ... 72	\glslinkvar: new ..... 92
\gls@istfilebase: new ..... 33	\GLSname: switched to using \@gls@hyp@opt ..... 116
\glsaddkey: removed \@sGLS@user@⟨key⟩ ..... 69	\Glsname: switched to using \@gls@hyp@opt ..... 115
removed \@sGls@user@⟨key⟩ . 68	\glsname: switched to using \@gls@hyp@opt ..... 115
removed \@sgls@user@⟨key⟩ . 68	\GLSpl: switched to using \@gls@hyp@opt ..... 110
switched to using \@gls@hyp@opt ..... 68, 69	\Glspl: switched to using \@gls@hyp@opt ..... 109
\GLSdesc: switched to using \@gls@hyp@opt ..... 117	\glspl: switched to using \@gls@hyp@opt ..... 108
\Glsdesc: switched to using \@gls@hyp@opt ..... 116	\GLSplural: switched to using \@gls@hyp@opt ..... 114
\glsdesc: switched to using \@gls@hyp@opt ..... 116	\Glsplural: switched to using \@gls@hyp@opt ..... 114
\GLSdescplural: switched to us- ing \@gls@hyp@opt ..... 118	\glsplural: switched to using \@gls@hyp@opt ..... 113
\Glsdescplural: switched to us- ing \@gls@hyp@opt ..... 117	\glsspace: new ..... 194
\glsdescplural: switched to us- ing \@gls@hyp@opt ..... 117	\GLSsymbol: switched to using \@gls@hyp@opt ..... 118
\glsdisablehyper: added \KV@glslink@hyperfalse to definition ..... 105	\Glsymbol: switched to using \@gls@hyp@opt ..... 118
\glsdisp: switched to using \@gls@hyp@opt ..... 111	\glssymbol: switched to using \@gls@hyp@opt ..... 118
\glsdohyperlink: new ..... 104	\GLSsymbolplural: switched to using \@gls@hyp@opt .... 119
\glsdohypertarget: new .... 104	\Glsymbolplural: switched to using \@gls@hyp@opt .... 119
\glsenablehyper: added \KV@glslink@hypertrue to definition ..... 105	\glssymbolplural: switched to using \@gls@hyp@opt .... 119
\GLSfirst: switched to using \@gls@hyp@opt ..... 113	\GLStext: switched to using \@gls@hyp@opt ..... 112
\Glsfirst: switched to using \@gls@hyp@opt ..... 113	\Glstext: switched to using \@gls@hyp@opt ..... 112
\glsfirst: switched to using \@gls@hyp@opt ..... 112	\glstext: switched to using \@gls@hyp@opt ..... 112
\GLSfirstplural: switched to using \@gls@hyp@opt .... 115	\glstreenamfmt: new ..... 280
\Glsfirstplural: switched to using \@gls@hyp@opt .... 115	\GLSuseri: switched to using \@gls@hyp@opt ..... 120
\glsfirstplural: switched to using \@gls@hyp@opt .... 114	\Glsuseri: switched to using \@gls@hyp@opt ..... 120
\glsifhyper: deprecated ..... 92	\glsuseri: switched to using \@gls@hyp@opt ..... 120

\GLSuserii: switched to using \@gls@hyp@opt .....	121	\ifignoredglossary:new ....	57
\Glsuserii: switched to using \@gls@hyp@opt .....	121	altlongragged4col: fixed bug that displayed description in- stead of symbol .....	261
\glsuserii: switched to using \@gls@hyp@opt .....	121	\newglossary: added starred ver- sion .....	54
\GLSuseriii: switched to using \@gls@hyp@opt .....	122	\newignoredglossary:new ...	57
\Glsuseriii: switched to using \@gls@hyp@opt .....	122	\ns@newglossary: added \@glotype@<name>@log ...	55
\glsuseriii: switched to using \@gls@hyp@opt .....	121	new .....	55
\GLSuseriv: switched to using \@gls@hyp@opt .....	123	\p@gls@hyp@opt:new .....	93
\Glsuseriv: switched to using \@gls@hyp@opt .....	123	\PGLS: changed to use \@gls@hyp@opt .....	239
\glsuseriv: switched to using \@gls@hyp@opt .....	122	\PglS: changed to use \@gls@hyp@opt .....	238
\GLSuserv: switched to using \@gls@hyp@opt .....	124	\pgls: changed to use \@gls@hyp@opt .....	236
\Glsuserv: switched to using \@gls@hyp@opt .....	124	\PGLSpl: changed to use \@gls@hyp@opt .....	240
\glsuserv: switched to using \@gls@hyp@opt .....	123	\PglSpl: changed to use \@gls@hyp@opt .....	238
\GLSuservi: switched to using \@gls@hyp@opt .....	125	\pglSpl: changed to use \@gls@hyp@opt .....	237
\Glsuservi: switched to using \@gls@hyp@opt .....	124	\s@gls@hyp@opt:new .....	92
\glsuservi: switched to using \@gls@hyp@opt .....	124	\s@newglossary:new .....	54
	4.09	automake:new .....	24
		\glsaddkey: fixed bug in user commands .....	68

## Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.

Symbols			
\@@do@@wrglossary	160	\@glo@addchildren	168
\@@do@wrglossary	159	\@glo@default@sorttype	9
\@@glo@assign@sortkey	180	\@glo@do@sortentries	169
\@@glossarysec	5	\@glo@grabfirst	174
\@@glossaryseclabel	6	\@glo@no@assign@sortkey	180
\@@glossarysecstar	6	\@glo@seeautonumberlist	7
\@@gls@default@entryfmt	315	\@glo@sortedinsert	169
\@@gls@expand@field	63	\@glo@sortentries	168
\@@gls@fixbraces	162	\@glo@sorthandler@case	170
\@@gls@noidx@nosanitizesort	18	\@glo@sorthandler@letter	170
\@@gls@noidx@sanitizesort	18	\@glo@sorthandler@nocase	170
\@@gls@nosanitizesort	18	\@glo@sorthandler@word	170
\@@gls@sanitizesort	18	\@glo@sortmacro@case	171
\@ACRlong	326	\@glo@sortmacro@def	172
\@ACRshort	325	\@glo@sortmacro@def@do	172
\@Acrlong	326	\@glo@sortmacro@letter	171
\@Acrshort	325	\@glo@sortmacro@nocase	172
\@GLS@	108	\@glo@sortmacro@standard	171
\@GLSpl	110	\@glo@sortmacro@use	173
\@Gls@	107	\@glo@sortmacro@word	170
\@Gls@entry@field	133	\@glo@storeentry	76
\@Glspl@	109	\@glo@types	53
\@PGLS	239	\@glossary	157
\@PGLS@	239	\@glossary@default@style	6
\@PGLSpl	240	\@glossaryentryfield	76
\@PGLSpl@	240	\@glossarysection	37
\@Pgls	238	\@glossarysubentryfield	76
\@Pgls@	238	\@gls	106
\@Pglspl	238	\@gls@	106
\@Pglspl@	239	\@gls@@link	93
\@acrfull	194	\@gls@access@display	313
\@acrlong	326	\@gls@addpredefinedattributes	
\@acrshort	325		41
\@addtoacronymlists	14	\@gls@assign@symbol@field	18
\@closegls	149	\@gls@assign@symbolplural@field	
\@delimN	191		18
\@delimR	190	\@gls@checkactual	102
\@disable@onlypremakeg	29	\@gls@checkboxar	101
\@disable@premakecs	29	\@gls@checkescactual	100
\@disabled@glsaddxdycounters	40	\@gls@checkescbar	100
\@do@seeglossary	161	\@gls@checkesclevel	101
\@do@wrglossary	158, 289	\@gls@checkescquote	99
		\@gls@checklevel	102

<code>\@gls@checkmkidxchars</code>	98	<code>\@gls@saveentrycounter</code>	95
<code>\@gls@checkquote</code>	99	<code>\@gls@setacrstyle</code>	22
<code>\@gls@codepage</code>	47	<code>\@gls@setcounter</code>	56
<code>\@gls@counterwithin</code>	9	<code>\@gls@setupsort@def</code>	11
<code>\@gls@declareoption</code>	7	<code>\@gls@setupsort@standard</code>	10
<code>\@gls@default@value</code>	59	<code>\@gls@setupsort@use</code>	11
<code>\@gls@do@acronymsdef</code>	13	<code>\@gls@startswithexpandonce</code>	64
<code>\@gls@doautomake</code>	25	<code>\@gls@symbolsdef</code>	26
<code>\@gls@entry@field</code>	132	<code>\@gls@tmpb</code>	99
<code>\@gls@escbsdq</code>	97	<code>\@gls@toc</code>	38
<code>\@gls@expand@fields</code>	63	<code>\@gls@updatechecked</code>	99
<code>\@gls@fetchfield</code>	66	<code>\@gls@warnonglossdefined</code>	16
<code>\@gls@field@link</code>	111	<code>\@gls@warnontheGLOSSdefined</code>	16
<code>\@gls@fixbraces</code>	162	<code>\@gls@writedef</code>	65
<code>\@gls@forbidtext</code>	54	<code>\@gls@xdy@Lclass@Alpha-page-numbers</code>	43
<code>\@gls@getcounter</code>	56	<code>\@gls@xdy@Lclass@Appendix-page-numbers</code>	43
<code>\@gls@getcounterprefix</code>	161	<code>\@gls@xdy@Lclass@Roman-page-numbers</code>	43
<code>\@gls@getgrouptitle</code>	187	<code>\@gls@xdy@Lclass@alpha-page-numbers</code>	43
<code>\@gls@getothergrouptitle</code>	187	<code>\@gls@xdy@Lclass@arabic-page-numbers</code>	43
<code>\@gls@hyp@opt</code>	92	<code>\@gls@xdy@Lclass@arabic-section-numbers</code>	44
<code>\@gls@hypergroup</code>	245	<code>\@gls@xdy@Lclass@roman-page-numbers</code>	43
<code>\@gls@ifinlist</code>	39	<code>\@gls@xdy@locationlist</code>	42
<code>\@gls@indexdef</code>	26	<code>\@gls@xdy@checkbackslash</code>	103
<code>\@gls@keymap</code>	66, 234	<code>\@gls@xdy@checkquote</code>	103
<code>\@gls@link</code>	94	<code>\@gls@Alphacompositor</code>	33, 43
<code>\@gls@link@checkfirsthyper</code>	93	<code>\@gls@acronymlists</code>	14
<code>\@gls@loadlist</code>	8	<code>\@gls@defaultplural</code>	62
<code>\@gls@loadlong</code>	8	<code>\@gls@defaultsort</code>	62
<code>\@gls@loadsuper</code>	8	<code>\@gls@disp</code>	111
<code>\@gls@loadtree</code>	8	<code>\@gls@firstletter</code>	141
<code>\@gls@makefirsttuc</code>	241	<code>\@gls@hypernumber</code>	190
<code>\@gls@missingnumberlist</code>	62	<code>\@gls@link</code>	104
<code>\@gls@noaccess</code>	310	<code>\@gls@minrange</code>	142
<code>\@gls@noexpand@field</code>	62	<code>\@gls@nextpages</code>	180
<code>\@gls@noexpand@fields</code>	62	<code>\@gls@nodesc</code>	62
<code>\@gls@nohyperlist</code>	16	<code>\@gls@noname</code>	61
<code>\@gls@noidx@do</code>	174	<code>\@gls@nonextpages</code>	180
<code>\@gls@noidx@setsanitizesort</code>	20	<code>\@gls@openfile</code>	148
<code>\@gls@noref@warn</code>	155	<code>\@gls@order</code>	23
<code>\@gls@notranslatorhook</code>	21	<code>\@gls@spl@</code>	108
<code>\@gls@numbersdef</code>	26	<code>\@gls@target</code>	105
<code>\@gls@onlypremakeg</code>	28	<code>\@gls@widestname</code>	284
<code>\@gls@provide@newglossary</code>	53		
<code>\@gls@reference</code>	176		
<code>\@gls@renewglossary</code>	157		
<code>\@gls@sanitizedesc</code>	17		
<code>\@gls@sanitizename</code>	17		
<code>\@gls@sanitizesort</code>	18		
<code>\@gls@sanitizesymbol</code>	18		



sm-short-long ..... 202  
 sm-short-long-desc .. 203, 331  
 \acronymentry ..... 198  
 \acronymfont .....  
     ..... 87, 197, 214, 218, 221, 224  
 acronymlists (option) ..... 15  
 \acronymname ..... 29  
 acronyms (option) ..... 14  
 \acronymsort ..... 198  
 \acronymtype ..... 13, 193  
 \acrpluralsuffix ..... 193  
 \ACRshort ..... 126  
 \Acrshort ..... 126  
 \acrshort ..... 125  
 \ACRshortpl ..... 128  
 \Acrshortpl ..... 127  
 \acrshortpl ..... 127  
 \Acs ..... 209  
 \acs ..... 209  
 \Acsp ..... 209  
 \acsp ..... 209  
 \addglossarytocaptions ..... 30  
 \addto ..... 30  
 align (environment) ..... 78, 95, 96  
 altlist (style) ..... 250  
 altlistgroup (style) ..... 250  
 altlisthypergroup (style) ..... 250  
 altlong4col (style) ..... 256  
 altlong4colborder (style) ..... 257  
 altlong4colheader (style) ..... 257  
 altlong4colheaderborder (style) 257  
 altlongragged4col (style) ..... 261  
 altlongragged4colborder (style) 262  
 altlongragged4colheader (style) 262  
 altlongragged4colheaderborder  
     (style) ..... 263  
 \altnewglossary ..... 56  
 altsuper4col (style) ..... 272  
 altsuper4colborder (style) .... 273  
 altsuper4colheader (style) .... 273  
 altsuper4colheaderborder  
     (style) ..... 273  
 altsuperragged4col (style) .... 278  
 altsuperragged4colborder  
     (style) ..... 279  
 altsuperragged4colheader  
     (style) ..... 278  
 altsuperragged4colheaderborder  
     (style) ..... 279

alttree (style) ..... 284  
 alttreegroup (style) ..... 287  
 alttreehypergroup (style) ..... 287  
 amsgen package ..... 4, 91  
 amsmath package ..... 78  
 \andname ..... 30  
 array package ..... 258, 274  
 article class ..... 161  
 automake (option) ..... 24

## B

babel package ..... 28, 29, 31, 46, 344

## C

\capitalisewords ..... 242  
 \changes ..... 5  
 \compatglossarystyle ..... 294  
 compatible-2.07 (option) ..... 25  
 compatible-3.07 (option) ..... 25  
 \compatibleglossentry .. 183, 307  
 \compatiblesubglossentry 185, 308  
 counter (key) ..... 59  
 counter (option) ..... 15  
 \CustomAcronymFields ..... 226  
 \CustomNewAcronymDef ..... 227

## D

datatool package ..... 169  
 \DeclareAcronymList ..... 14  
 \DefaultNewAcronymDef .. 211, 337  
 \defentryfmt ..... 91  
 \defglstdisplay ..... 90  
 \defglstdisplayfirst ..... 90  
 \defglssentry ..... 55  
 \defglssentryfmt ..... 54, 58, 59, 82  
 \DefineAcronymSynonyms ..... 209  
 \delimN ..... 35, 190  
 \delimR ..... 35, 190  
 description (environment) 248, 249  
 description (key) ..... 58  
 description (option) ..... 22  
 descriptionaccess (key) ..... 309  
 \DescriptionDUANewAcronymDef 215  
 \DescriptionFootnoteNewAcronymDef  
     ..... 213, 339  
 \descriptionname ..... 30  
 \DescriptionNewAcronymDef ..  
     ..... 217, 339  
 descriptionplural (key) ..... 58  
 descriptionpluralaccess (key) 309

\detokenize ..... 49  
doc package ..... 4, 5, 12  
dua (acrstyle) ..... 204, 331  
dua (option) ..... 23  
dua-desc (acrstyle) ..... 206, 334  
\DUANewAcronymDef ..... 224

## E

entrycounter (option) ..... 9  
entrycounterwithin (option) .... 9  
\entryname ..... 29  
environments:  
    align ..... 78, 95, 96  
    description ..... 248, 249  
    longtable ..... 8, 228, 251–263  
    multicols ..... 263  
    supertabular ... 8, 228, 267–279  
    theglossary ..... 5,  
        16, 35, 183, 189, 265, 266, 282–285  
    theindex ..... 280  
equation (counter) ..... 95, 96  
etoolbox package ..... 4, 240

## F

file types  
    .aux ..... 166  
    .glo ..... 77  
    .ist ..... 141, 147, 148  
    .toc ..... 38  
    .xdy ..... 33  
    glo ..... 234  
\findrootlanguage ..... 46  
first (key) ..... 59  
firstaccess (key) ..... 309  
\firstacronymfont ..... 87, 197  
firstplural (key) ..... 59  
firstpluralaccess (key) ..... 309  
footnote (acrstyle) ..... 206, 334  
footnote (option) ..... 22  
footnote-desc (acrstyle) .. 208, 336  
footnote-sc (acrstyle) .... 207, 336  
footnote-sc-desc (acrstyle) 208, 337  
footnote-sm (acrstyle) .... 207, 336  
footnote-sm-desc (acrstyle) 208, 337  
\FootnoteNewAcronymDef . 219, 340  
\forallacronyms ..... 48  
\forallglossaries ..... 48  
\forallglsentries ..... 48  
\forglsentries ..... 48

## G

garamondx package ..... 193  
\Genacrfullformat ..... 89, 324  
\genacrfullformat ..... 89, 324  
\GenericAcronymFields ..... 198  
\Genplacrfullformat ..... 90, 324  
\genplacrfullformat ..... 89, 324  
\glo@grabfirst ..... 173  
\glolinkprefix ..... 95  
glossareentry (counter) ..... 181  
glossaries package ..... 26,  
    46, 47, 142, 228, 234, 248, 287, 307  
glossaries-accsupp package ... 76, 307  
\GlossariesWarning ..... 16  
\GlossariesWarningNoLine .... 16  
\glossary ..... 54, 148, 157, 188  
glossary counters:  
    glossaryentry ..... 180  
    glossarysubentry ..... 181  
glossary keys:  
    access ..... 308  
    counter ..... 59  
    description ..... 58  
    descriptionaccess ..... 309  
    descriptionplural ..... 58  
    descriptionpluralaccess . 309  
    first ..... 59  
    firstaccess ..... 309  
    firstplural ..... 59  
    firstpluralaccess ..... 309  
    long ..... 61  
    longaccess ..... 309  
    longplural ..... 61  
    longpluralaccess ..... 310  
    name ..... 58  
    nonumberlist ..... 60  
    parent ..... 60  
    plural ..... 58  
    pluralaccess ..... 309  
    see ..... 60  
    short ..... 61  
    shortaccess ..... 309  
    shortplural ..... 61  
    shortpluralaccess ..... 309  
    sort ..... 58  
    symbol ..... 59  
    symbolaccess ..... 309  
    symbolplural ..... 59  
    symbolpluralaccess ..... 309



text .....	58	altsuperragged4colborder	
textaccess .....	308	.....	279, 305
type .....	59	altsuperragged4colborder	279
user1 .....	60	altsuperragged4colheader	
user2 .....	60	.....	278, 305
user3 .....	61	altsuperragged4colheader	278
user4 .....	61	altsuperragged4colheaderborder	
user5 .....	61	.....	279, 305
user6 .....	61	altsuperragged4colheaderborder	
glossary package .....	1, 192	.....	279
glossary styles:		alttree .....	266, 284, 287, 301
altlist .....	250, 295	alttree .....	284
altlist .....	250	alttreegroup .....	287, 303
altlistgroup .....	250, 296	alttreegroup .....	287
altlistgroup .....	250	alttreehypergroup ...	287, 303
altlisthypergroup ...	250, 296	alttreehypergroup .....	287
altlisthypergroup .....	250	index .....	263, 280, 281, 299
altlong4col ..	256, 257, 261, 298	index .....	280
altlong4col .....	256	indexgroup .....	281, 300
altlong4colborder ...	257, 298	indexgroup .....	281
altlong4colborder .....	257	indexhypergroup .....	281, 300
altlong4colheader ...	257, 298	indexhypergroup .....	281
altlong4colheader .....	257	inline .....	294
altlong4colheaderborder .		inline .....	246
.....	257, 298	list .....	6, 248–251, 295
altlong4colheaderborder .	257	list .....	248
altlongragged4col	261, 262, 299	listdotted .....	251, 296
altlongragged4col .....	261	listdotted .....	251
altlongragged4colborder .		listgroup .....	249, 295
.....	262, 299	listgroup .....	249
altlongragged4colborder .	262	listhypergroup .....	249, 295
altlongragged4colheader .		listhypergroup .....	249
.....	262, 299	long .....	252, 253, 258, 296, 298
altlongragged4colheader .	262	long .....	252
altlongragged4colheaderborder		long3col .....	253, 254, 297
.....	263, 299	long3col .....	253
altlongragged4colheaderborder		long3colborder .....	254, 297
.....	263	long3colborder .....	254
altsuper4col .	272, 273, 278, 307	long3colheader .....	254, 297
altsuper4col .....	272	long3colheader .....	254
altsuper4colborder ..	273, 307	long3colheaderborder	254, 297
altsuper4colborder .....	273	long3colheaderborder ...	254
altsuper4colheader ..	273, 307	long4col .....	255, 256, 297
altsuper4colheader .....	273	long4col .....	255
altsuper4colheaderborder		long4colborder .....	256, 297
.....	273, 307	long4colborder .....	256
altsuper4colheaderborder	273	long4colheader .....	255, 297
altsuperragged4col	278, 279, 305	long4colheader .....	255
altsuperragged4col .....	278	long4colheaderborder	256, 297

long4colheaderborder	....	256
longborder	.....	252, 296
longborder	.....	252
longheader	.....	253, 296
longheader	.....	253
longheaderborder	....	253, 296
longheaderborder	.....	253
longragged	.....	258–260
longragged	.....	258
longragged3col	...	260, 261, 298
longragged3col	.....	260
longragged3colborder		260, 299
longragged3colborder	....	260
longragged3colheader		261, 299
longragged3colheader	....	261
longragged3colheaderborder	.....	261, 299
longragged3colheaderborder	.....	261
longraggedborder	....	259, 298
longraggedborder	.....	259
longraggedheader	....	259, 298
longraggedheader	.....	259
longraggedheaderborder		259, 298
longraggedheaderborder	..	259
mcolalttree	.....	266, 304
mcolalttree	.....	266
mcolalttreegroup	....	266, 304
mcolalttreegroup	.....	266
mcolalttreehypergroup		266, 304
mcolalttreehypergroup	...	266
mcolindex	.....	264, 303
mcolindex	.....	263
mcolindexgroup	.....	264, 303
mcolindexgroup	.....	264
mcolindexhypergroup	.	264, 303
mcolindexhypergroup	....	264
mcoltree	.....	264, 303
mcoltree	.....	264
mcoltreegroup	.....	303
mcoltreegroup	.....	264
mcoltreehypergroup	..	265, 303
mcoltreehypergroup	.....	265
mcoltreenoname	.....	265, 303
mcoltreenoname	.....	265
mcoltreenonamegroup	.	265, 303
mcoltreenonamegroup	....	265
mcoltreenonamehypergroup	.....	265, 303
mcoltreenonamehypergroup		265
sublistdotted	.....	296
sublistdotted	.....	251
super	.....	267–269, 275, 305
super	.....	267
super3col	.....	269, 270, 306
super3col	.....	269
super3colborder	....	270, 306
super3colborder	.....	270
super3colheader	....	270, 306
super3colheader	.....	270
super3colheaderborder		270, 306
super3colheaderborder	...	270
super4col	.....	271, 272, 306
super4col	.....	271
super4colborder	....	272, 307
super4colborder	.....	272
super4colheader	....	271, 306
super4colheader	.....	271
super4colheaderborder		272, 307
super4colheaderborder	...	272
superborder	.....	268, 305
superborder	.....	268
superheader	.....	268, 306
superheader	.....	268
superheaderborder	...	269, 306
superheaderborder	.....	269
superragged	.....	274, 276, 304
superragged	.....	274
superragged3col	..	276, 277, 304
superragged3col	.....	276
superragged3colborder		277, 305
superragged3colborder	...	277
superragged3colheader		277, 305
superragged3colheader	...	277
superragged3colheaderborder	.....	277, 305
superraggedborder	...	275, 304
superraggedborder	.....	275
superraggedheader	...	275, 304
superraggedheader	.....	275
superraggedheaderborder	.	
superraggedheaderborder	.....	276, 304
superraggedheaderborder	.	276
superraggedright3colheaderborder	.....	277
tree	.....	264, 281–284, 300
tree	.....	281
treegroup	.....	265, 282, 301

treegroup .....	282	\gls@assign@desc .....	69
treehypergroup .....	282, 301	\gls@assign@desc@field .....	17
treehypergroup .....	282	\gls@assign@descplural@field .....	17
treenoname ...	265, 283, 284, 301	\gls@assign@field .....	64
treenoname .....	283	\gls@assign@name@field .....	17
treenonamegroup .....	284, 301	\gls@assign@type@field .....	17
treenonamegroup .....	284	\gls@checkisacronymlist .....	15
treenonamehypergroup .....	284, 301	\gls@checkseeallowed .....	60
treenonamehypergroup .....	284	\gls@codepage .....	24
glossary-hypernav package .....	141	\gls@defglossaryentry .....	70
glossary-list package .....	6, 8, 248	\gls@disablepagerefexpansion .....	158
glossary-long package .....		\gls@docclearpage .....	38
.....	8, 251, 252, 261, 267	\gls@doentryfmt .....	54
glossary-longragged package .....	258	\gls@hypergroup prerun .....	245
glossary-mcols package .....	263	\gls@ifnotmeasuring .....	78
glossary-super package .....		\gls@istfilebase .....	33
.....	8, 252, 267, 274, 278	\gls@level .....	62
glossary-superragged package .....	274	\gls@noidxglossary .....	156
glossary-tree package .....	8, 280	\gls@numberpage .....	158
glossaryentry (counter) ..	9, 181, 182	\gls@protected@pagefmts .....	158
glossaryentry (counter) .....	180	\gls@Romanpage .....	159
\glossaryentryfield .....	185, 189	\gls@romanpage .....	159
\glossaryentrynumber .....	180	\gls@save@numberlist .....	163
\glossaryentrynumbers .....		\gls@suffixF .....	34
.....	7, 34, 165, 166	\gls@suffixFF .....	34
\glossaryheader .....	183, 189	\glsaccessdisplay .....	315
\glossarymark .....	36	\glsaccsupp .....	313
\glossaryname .....	29, 30	\glsadd .....	80, 139, 140, 188
\glossarypostamble .....	36, 189	\glsadd options	
\glossarypreamble .....	35, 189	counter .....	140
\glossarysection .....	6, 36, 54	format .....	140, 189
\glossarystyle .....	188, 228	\glsaddall .....	49, 80, 140
glossarysubentry (counter) ...		\glsaddall options	
.....	9, 181, 182	types .....	140
glossarysubentry (counter) ...	181	\glsaddallunused .....	140
\glossarysubentryfield .....	185	\glsaddkey .....	67
\glossentry .....	59, 183	\GlsAddLetterGroup .....	47
\Glossentrydesc .....	184	\GlsAddSortRule .....	45
\glossentrydesc .....	184, 327	\GlsAddXdyAlphabet .....	42
\Glossentryname .....	184	\GlsAddXdyAttribute .....	40, 287
\glossentryname .....	184, 327	\GlsAddXdyCounters .....	39, 288
\Glossentrysymbol .....	184	\GlsAddXdyLocation .....	44, 288
\glossentrysymbol .....	184, 327	\GlsAddXdyStyle .....	46
\GLS .....	107	\glsautoprefix .....	6
\Gls .....	107, 109, 240	\glsclearpage .....	38
\gls .....	4, 59,	\glsclosebrace .....	141
	80, 92, 106–108, 112–124, 182, 236	\glscompositor .....	33, 44
\gls@Alphpage .....	158	\glscounter .....	15, 56
\gls@alphpage .....	158	\GlsDeclareNoHyperList .....	16

<code>\glsdefaulttype</code> .....	13	<code>\Glsentrylongpl</code> .....	137
<code>\glsdefmain</code> .....	12	<code>\glsentrylongpl</code> .....	137
<code>\GLSdesc</code> .....	117	<code>\glsentrylongpluralaccess</code> ..	313
<code>\Glsdesc</code> .....	116	<code>\Glsentryname</code> .....	133
<code>\glsdesc</code> .....	116, 117	<code>\glsentryname</code> .....	133, 163
<code>\GLSdescplural</code> .....	118	<code>\glsentrynumberlist</code> ....	138, 153
<code>\Glsdescplural</code> .....	117	<code>\Glsentryplural</code> .....	134
<code>\glsdescplural</code> .....	117, 118	<code>\glsentryplural</code> .....	134
<code>\glsdescriptionaccessdisplay</code>	314	<code>\glsentrypluralaccess</code> .....	312
<code>\glsdescriptionpluralaccessdisplay</code>	314	<code>\Glsentryprefix</code> .....	236
.....	314	<code>\glsentryprefix</code> .....	235
<code>\glsdescwidth</code> ...	252, 258, 267, 274	<code>\Glsentryprefixfirst</code> .....	235
<code>\glsdetoklabel</code> .....	49	<code>\glsentryprefixfirst</code> .....	235
<code>\glsdisablehyper</code> .....	105	<code>\Glsentryprefixfirstplural</code> .	235
<code>\glsdisp</code> .....	111	<code>\glsentryprefixfirstplural</code> .	235
<code>\glsdisplay</code> .....	81, 90, 106	<code>\Glsentryprefixplural</code> .....	236
<code>\glsdisplayfirst</code> .....	81, 90, 106	<code>\glsentryprefixplural</code> .....	235
<code>\glsdisplaynumberlist</code>	138, 154, 176	<code>\Glsentryshort</code> .....	137
<code>\glsdohyperlink</code> .....	104	<code>\glsentryshort</code> .....	137
<code>\glsdohypertarget</code> .....	104	<code>\glsentryshortaccess</code> .....	313
<code>\glsdoifexists</code> .....	50	<code>\Glsentryshortpl</code> .....	137
<code>\glsdoifexistsorwarn</code> .....	50	<code>\glsentryshortpl</code> .....	137
<code>\glsdoifnoexists</code> .....	50	<code>\glsentryshortpluralaccess</code> .	313
<code>\glsdoparenifnotempty</code> .....	221	<code>\glsentrysort</code> .....	135
<code>\glsenablehyper</code> .....	105	<code>\Glsentrysymbol</code> .....	134
<code>\glsentryaccess</code> .....	312	<code>\glsentrysymbol</code> .....	134
<code>\glsentrycounter</code> .....	95	<code>\glsentrysymbolaccess</code> .....	312
<code>\glsentrycounterlabel</code> .....	182	<code>\Glsentrysymbolplural</code> .....	135
<code>\Glsentrydesc</code> .....	133	<code>\glsentrysymbolplural</code> .....	134
<code>\glsentrydesc</code> .....	133	<code>\glsentrysymbolpluralaccess</code>	312
<code>\glsentrydescaccess</code> .....	312	<code>\Glsentrytext</code> .....	134
<code>\Glsentrydescplural</code> .....	134	<code>\glsentrytext</code> .....	49, 134, 163
<code>\glsentrydescplural</code> .....	133	<code>\glsentrytextaccess</code> .....	312
<code>\glsentrydescpluralaccess</code> ..	312	<code>\glsentrytype</code> .....	135
<code>\Glsentryfirst</code> .....	135	<code>\Glsentryuseri</code> .....	136
<code>\glsentryfirst</code> .....	135	<code>\glsentryuseri</code> .....	135
<code>\glsentryfirstaccess</code> .....	312	<code>\Glsentryuserii</code> .....	136
<code>\Glsentryfirstplural</code> .....	135	<code>\glsentryuserii</code> .....	136
<code>\glsentryfirstplural</code> .....	135	<code>\Glsentryuseriii</code> .....	136
<code>\glsentryfirstpluralaccess</code> .	312	<code>\glsentryuseriii</code> .....	136
<code>\glsentryfmt</code> .....	58, 59, 81	<code>\Glsentryuseriv</code> .....	136
<code>\Glsentryfull</code> .....	138	<code>\glsentryuseriv</code> .....	136
<code>\glsentryfull</code> ...	138, 198, 207, 336	<code>\Glsentryuserv</code> .....	136
<code>\Glsentryfullpl</code> .....	138	<code>\glsentryuserv</code> .....	136
<code>\glsentryfullpl</code> .....	138	<code>\Glsentryuservi</code> .....	137
<code>\glsentryitem</code> .....	182	<code>\glsentryuservi</code> .....	137
<code>\Glsentrylong</code> .....	137	<code>\glsexpandfields</code> .....	64
<code>\glsentrylong</code> .....	137	<code>\GLSfirst</code> .....	113
<code>\glsentrylongaccess</code> .....	313	<code>\Glsfirst</code> .....	113

<code>\glsfirst</code> .....	112, 113	<code>\glslongpluralaccessdisplay</code>	315
<code>\glsfirstaccessdisplay</code> .....	314	<code>\glslongpluralaccesskey</code> ....	342
<code>\GLSfirstplural</code> .....	115	<code>\glslongpluralkey</code> .....	194
<code>\Glsfirstplural</code> .....	115	<code>\glslongtok</code> .....	197
<code>\glsfirstplural</code> .....	114, 115	<code>\glsmakefirstuc</code> .....	242
<code>\glsfirstpluralaccessdisplay</code>	314	<code>\glsncols</code> .....	263
<code>\glsgenacfmt</code> .....	87, 322	<code>\glsmoveentry</code> .....	76
<code>\glsgenentryfmt</code> .....	85, 319	<code>\GLSname</code> .....	116
<code>\glsgetgrouplabel</code> .....	188	<code>\Glsname</code> .....	115
<code>\glsgetgrouptitle</code> .....	141, 187	<code>\glsname</code> .....	115, 116
<code>\glsglossarymark</code> .....	9, 36	<code>\glsnameaccessdisplay</code> .....	313
<code>\glsgroupheading</code> .....	186, 189	<code>\glsnamefont</code> .....	189, 280
<code>\glsgroupskip</code> .....	186, 189, 248	<code>\glsnavhyperlink</code> .....	244
<code>\glshyperlink</code> .....	139	<code>\glsnavhypertarget</code> .....	244
<code>\glshypernavsep</code> .....	246	<code>\glsnavigation</code> .....	245
<code>\glshypernumber</code> .....	34, 190	<code>\glsnextpages</code> .....	180
<code>\glsifhyper</code> .....	92	<code>\glsnoexpandfields</code> .....	64
<code>\glsifhyperon</code> .....	92	<code>\glsnoidxdisplayloc</code> .....	176
<code>\glsIfListOfAcronyms</code> .....	14	<code>\glsnoidxdisplaylocclsthander</code>	
<code>\glsinlinedescformat</code> .....	248	.....	176
<code>\glsinlinedopostchild</code> ..	246, 247	<code>\glsnoidxloclist</code> .....	175
<code>\glsinlineemptydescformat</code> ..	248	<code>\glsnoidxloclisthandler</code> ....	176
<code>\glsinlinenameformat</code> .....	248	<code>\glsnoidxnumberlistloophandler</code>	
<code>\glsinlineparentchildseparator</code>		.....	155
.....	248	<code>\glsnoidxstripaccents</code> .....	19
<code>\glsinlinepostchild</code> .....	248	<code>\glsnonextpages</code> .....	180
<code>\glsinlineseparator</code> .....	248	<code>\glsnoxindywarning</code> .....	39
<code>\glsinlinesubdescformat</code> ....	248	<code>\glsnumberformat</code> .....	34
<code>\glsinlinesubnameformat</code> ....	248	<code>\glsnumberlistloop</code> .....	155
<code>\glsinlinesubseparator</code> ....	248	<code>\glsnumbersgroupname</code> .	30, 141, 186
<code>\glskeylisttok</code> .....	197	<code>\glsnumlistlastsep</code> .....	139
<code>\glslabeltok</code> .....	197	<code>\glsnumlistsep</code> .....	139
<code>\glsletentryfield</code> .....	132	<code>\glsopenbrace</code> .....	141
<code>\glslink</code> 80, 81, 93, 106, 139, 188, 189		<code>\glsorder</code> .....	23
<code>\glslink options</code>		<code>\glsorg@endtheglossary</code> .....	5
counter .....	91, 106, 234	<code>\glsorg@glossary</code> .....	4
format .....	91, 106, 189	<code>\glsorg@theglossary</code> .....	5
hyper .....	92, 93, 106	<code>\glsorg@wrglossary</code> .....	4
local .....	92	<code>\glspagelistwidth</code> 252, 258, 267, 274	
<code>\glslinkcheckfirsthyperhook</code> .	94	<code>\glspar</code> .....	32
<code>\glslinkvar</code> .....	92	<code>\GLSpl</code> .....	110
<code>\glslistdottedwidth</code> .....	251	<code>\Glspl</code> .....	109, 240
<code>\glslocalreset</code> .....	79	<code>\glspl</code> .....	80, 108–110
<code>\glslocalresetall</code> .....	79	<code>\GLSplural</code> .....	114
<code>\glslocalunset</code> .....	79	<code>\Glsplural</code> .....	114
<code>\glslocalunsetall</code> .....	80	<code>\glsplural</code> .....	113, 114
<code>\glslongaccessdisplay</code> .....	315	<code>\glspluralaccessdisplay</code> ....	314
<code>\glslongaccesskey</code> .....	342	<code>\glspluralsuffix</code> .....	30, 58, 59
<code>\glslongkey</code> .....	194	<code>\glspostdescription</code> .....	8

<code>\glspostinline</code>	248	<code>\Glssymbolplural</code>	119
<code>\glsprestandardsort</code>	10	<code>\glssymbolplural</code>	119
<code>\glsquote</code>	141	<code>\glssymbolpluralaccessdisplay</code>	314
<code>\glsrefentry</code>	182	.....	314
<code>\glsreset</code>	78	<code>\glssymbolsgroupname</code>	30, 141, 186
<code>\glsresetall</code>	79	<code>\glstarget</code>	183
<code>\glsresetentrylist</code>	180	<code>\GLStext</code>	112
<code>\glsresetsubentrycounter</code>	181	<code>\Glstext</code>	112
<code>\glssee</code>	162	<code>\glstext</code>	112
<code>\glsseeformat</code>	144, 162	<code>\glstextaccessdisplay</code>	314
<code>\glsseeitem</code>	163	<code>\glstextformat</code>	81
<code>\glsseeitemformat</code>	163	<code>\glstextup</code>	193
<code>\glsseelastsep</code>	163	<code>\glstreeindent</code>	282, 283
<code>\glsseelist</code>	162	<code>\glstreenamefmt</code>	280
<code>\glsseesep</code>	163	<code>\glsunset</code>	79
<code>\glsSetAlphaCompositor</code>	34	<code>\glsunsetall</code>	80
<code>\glsSetCompositor</code>	33	<code>\GlsUseAcrEntryDispStyle</code>	200
<code>\glssetexpandfield</code>	17	<code>\GlsUseAcrStyleDefs</code>	200
<code>\glssetnoexpandfield</code>	17	<code>\GLSuseri</code>	120
<code>\glsSetSuffixF</code>	34	<code>\Glsuseri</code>	120
<code>\glsSetSuffixFF</code>	34	<code>\glsuseri</code>	120
<code>\glssettoctitle</code>	29	<code>\GLSuserii</code>	121
<code>\glssetwidest</code>	284	<code>\Glsuserii</code>	121
<code>\GlsSetXdyCodePage</code>	47	<code>\glsuserii</code>	121
<code>\GlsSetXdyFirstLetterAfterDigits</code>	141	<code>\GLSuseriii</code>	122
.....	141	<code>\Glsuseriii</code>	122
<code>\GlsSetXdyLanguage</code>	46	<code>\glsuseriii</code>	121, 122
<code>\GlsSetXdyLocationClassOrder</code>	45	<code>\GLSuseriv</code>	123
<code>\GlsSetXdyMinRangeLength</code>	142	<code>\Glsuseriv</code>	123
<code>\GlsSetXdyStyles</code>	46	<code>\glsuseriv</code>	122, 123
<code>\glsshortaccessdisplay</code>	314	<code>\GLSuserv</code>	124
<code>\glsshortaccesskey</code>	342	<code>\Glsuserv</code>	124
<code>\glsshortkey</code>	193	<code>\glsuserv</code>	123, 124
<code>\glsshortpluralaccessdisplay</code>	314	<code>\GLSuservi</code>	125
<code>\glsshortpluralaccesskey</code>	342	<code>\Glsuservi</code>	124
<code>\glsshortpluralkey</code>	194	<code>\glsuservi</code>	124, 125
<code>\glsshorttok</code>	197	<code>\glswrite</code>	151
<code>\glssortnumberfmt</code>	10	<code>\glswritedefhook</code>	69
<code>\glsspace</code>	194	<code>\gMFUnocap</code>	243
<code>\glsstepentry</code>	181		
<code>\glsstepsubentry</code>	181		
<code>\glssubentrycounterlabel</code>	182		
<code>\glssubentryitem</code>	182		
<code>\GLSsymbol</code>	118		
<code>\Glssymbol</code>	118		
<code>\glssymbol</code>	118		
<code>\glssymbolaccessdisplay</code>	314		
<code>\glssymbolnav</code>	246		
<code>\GLSsymbolplural</code>	119		

## H

<code>\hyperbf</code>	192
<code>\hyperemph</code>	192
<code>hyperfirst (option)</code>	22
<code>\hyperit</code>	192
<code>\hyperlink</code>	104
<code>\hypermd</code>	192
<code>\hyperpage</code>	190
<code>hyperref package</code>	161, 164, 190, 234

`\hyperrm` ..... 191  
`\hypersc` ..... 192  
`\hypersf` ..... 191  
`\hypersl` ..... 192  
`\hypertarget` ..... 105  
`\hypertt` ..... 191  
`\hyperup` ..... 192

## I

`\if@gls@docloaded` ..... 4  
`\if@gls@isacronymlist` ..... 15  
`\ifglossaryexists` ..... 49  
`\ifglscdescsuppressed` ..... 51  
`\ifglsentryexists` ..... 49  
`\ifglshaschildren` ..... 50  
`\ifglshasdesc` ..... 51  
`\ifglshasfield` ..... 52  
`\ifglshaslong` ..... 51  
`\ifglshasparent` ..... 51  
`\ifglshasprefix` ..... 236  
`\ifglshasprefixfirst` ..... 236  
`\ifglshasprefixfirstplural` ..... 236  
`\ifglshasprefixplural` ..... 236  
`\ifglshasshort` ..... 52  
`\ifglshassymbol` ..... 51  
`\ifglstranslate` ..... 21  
`\ifglxsused` ..... 49, 78  
`\ifglxsindy` ..... 23  
`\ifignoredglossary` ..... 57  
`index (option)` ..... 26  
`index (style)` ..... 280  
`indexgroup (style)` ..... 281  
`indexhypergroup (style)` ..... 281  
`indexonlyfirst (option)` ..... 22  
`inline (style)` ..... 246  
`\inputencodingname` ..... 24  
`\istfile` ..... 156  
`\istfilename` ..... 33  
`\item` ..... 189, 248, 280

## L

`link text` ..... 81  
`list (style)` ..... 248  
`listdotted (style)` ..... 251  
`listgroup (style)` ..... 249  
`listhypergroup (style)` ..... 249  
`\loadglsentries` ..... 12, 80  
`long (key)` ..... 61  
`long (style)` ..... 252  
`long-sc-short (acrstyle)` ..... 201

`long-sc-short-desc (acrstyle)` ..  
 ..... 202, 330  
`long-short (acrstyle)` ..... 200, 328  
`long-short-desc (acrstyle)` . 202, 330  
`long-sm-short (acrstyle)` ..... 201  
`long-sm-short-desc (acrstyle)` ..  
 ..... 203, 330  
`long3col (style)` ..... 253  
`long3colborder (style)` ..... 254  
`long3colheader (style)` ..... 254  
`long3colheaderborder (style)` .. 254  
`long4col (style)` ..... 255  
`long4colborder (style)` ..... 256  
`long4colheader (style)` ..... 255  
`long4colheaderborder (style)` .. 256  
`longaccess (key)` ..... 309  
`longborder (style)` ..... 252  
`longheader (style)` ..... 253  
`longheaderborder (style)` ..... 253  
`\longnewglossaryentry` .... 58, 69  
`longplural (key)` ..... 61  
`longpluralaccess (key)` ..... 310  
`\longprovideglossaryentry` ... 70  
`longragged (style)` ..... 258  
`longragged3col (style)` ..... 260  
`longragged3colborder (style)` .. 260  
`longragged3colheader (style)` .. 261  
`longragged3colheaderborder`  
`(style)` ..... 261  
`longraggedborder (style)` ..... 259  
`longraggedheader (style)` ..... 259  
`longraggedheaderborder (style)` 259  
`longtable (environment)` .....  
 ..... 8, 228, 251–263  
`longtable package` ..... 251, 258

## M

`\makefirstuc` ..... 240  
`makeglossaries` .....  
 ..... 23, 33, 46, 47, 54, 150, 166  
`\makeglossaries` .....  
 ... 24, 28, 32–34, 53, 56, 150, 152  
`\makeglossary` ..... 152  
`makeindex` ..... 357  
`makeindex` ..... 10, 23, 24, 30,  
 33–35, 54, 56, 58, 77, 97, 100,  
 141, 144, 146, 148, 160, 186, 289  
`delim_n` ..... 34  
`delim_r` ..... 35



page\_compositor ..... 33  
 special characters ..... 98, 99, 141  
 makeindex (option) ..... 23  
 \makenoidxglossaries ..... 20, 152  
 mcolalmtree (style) ..... 266  
 mcolalmtreegroup (style) ..... 266  
 mcolalmtreehypergroup (style) . 266  
 mcolindex (style) ..... 263  
 mcolindexgroup (style) ..... 264  
 mcolindexhypergroup (style) ... 264  
 mcoltree (style) ..... 264  
 mcoltreegroup (style) ..... 264  
 mcoltreehypergroup (style) .... 265  
 mcoltreename (style) ..... 265  
 mcoltreenamegroup (style) ... 265  
 mcoltreenamehypergroup  
     (style) ..... 265  
 memoir class ..... 157  
 mfirstuc package ..... 1, 243  
 \mfirstucMakeUppercase ..... 242  
 \mfu@checkword ..... 242  
 \MFUclear ..... 243  
 \MFUnocap ..... 243  
 multicol package ..... 263  
 multicol (environment) ..... 263

## N

name (key) ..... 58  
 \new@glossaryentry ..... 65  
 \newacronym 22, 23, 61, 80, 81, 192, 193  
 \newacronymhook ..... 197  
 \newacronymstyle ..... 199  
 \newglossary 15, 54, 56, 148, 150, 177  
 \newglossaryentry 58, 64, 80, 81, 193  
 \newglossaryentry options  
   access ..... 310, 312  
   counter ..... 59  
   description ..... 22, 57, 58,  
     62, 64, 70, 116, 133, 193, 220, 309  
   descriptionaccess ..... 312, 314  
   descriptionplural ..... 117, 309  
   descriptionpluralaccess .. 312, 314  
   first ..... 59,  
     73, 105, 112, 135, 218, 223, 309  
   firstaccess ..... 312, 314  
   firstplural ..... 59, 114, 135, 309  
   firstpluralaccess ..... 312, 314  
   format ..... 143  
   long ..... 87, 137, 309

longaccess ..... 313, 315  
 longplural ..... 137, 310  
 longpluralaccess ..... 313, 315  
 name ..... 57,  
   58, 61, 64, 70, 115, 133, 163, 308  
 nonumberlist ..... 60  
 parent ..... 60, 64  
 plural ..... 58, 73, 113, 309  
 pluralaccess ..... 312, 314  
 prefix ..... 235  
 prefixfirst ..... 235  
 prefixfirstplural ..... 235  
 prefixplural ..... 235  
 see ..... 7, 60, 151, 152  
 short ..... 87, 137, 309  
 shortaccess ..... 313, 314  
 shortplural ..... 137, 309  
 shortpluralaccess ..... 313, 314  
 sort ..... 58, 135, 186  
 symbol ..... 58, 59, 118, 214,  
   216, 218, 223, 255, 271, 308–310  
 symbolaccess ..... 312, 314  
 symbolplural ..... 119, 309  
 symbolpluralaccess ..... 312, 314  
 text ..... 58,  
   59, 105, 112, 134, 214, 218, 308  
 textaccess ..... 312, 314  
 type ..... 12, 59, 80, 135  
 user1 ..... 120, 135, 310  
 user2 ..... 121, 136  
 user3 ..... 121, 136  
 user4 ..... 122, 136  
 user5 ..... 123, 136  
 user6 ..... 124, 137, 310  
 \newglossarystyle ..... 189  
 \newignoredglossary ..... 57  
 nogroupskip (option) ..... 9  
 nohypertypes (option) ..... 16  
 \noist ..... 148, 234, 294  
 nolist (option) ..... 8  
 nolong (option) ..... 8  
 nomain (option) ..... 13  
 nonumberlist (key) ..... 60  
 nonumberlist (option) ..... 7  
 \nopostdesc ..... 32  
 nopostdot (option) ..... 9  
 noredefwarn (option) ..... 16  
 nostyles (option) ..... 8  
 nosuper (option) ..... 8



notranslate (option)	21
notree (option)	8
nowarn (option)	16
\ns@newglossary	55
numberedsection (option)	6
numberline (option)	5
numbers (option)	26

## O

\oldacronym	192
order (option)	23

## P

\p@gl@s@hyp@opt	93
package options:	
acronym	13, 29, 164, 193
true	14
acronym	13
acronymlists	15
acronyms	14
automake	24
compatible-2.07	25
compatible-3.07	25
counter	15
counter	15
description	218, 219
description	22
dua	216, 218, 219
dua	23
entrycounter	179, 180
true	9
entrycounter	9
entrycounterwithin	9
footnote	106–111, 214, 216, 218, 220
footnote	22
hyperfirst	
false	106–111
hyperfirst	22
index	26
indexonlyfirst	364
indexonlyfirst	22
makeindex	144, 234
makeindex	23
nogroupskip	9
nohypertypes	16
nolist	228
nolist	8
nolong	228, 252
nolong	8
nomain	12, 13

nomain	13
nonumberlist	7
nonumberlist	7
nopostdot	9
noredefwarn	16
nostyles	8
nosuper	228
nosuper	8
notranslate	21
notree	228
notree	8
nowarn	16
numberedsection	6
numberline	5
numberline	5
numbers	26
order	23
sanitize	19, 58, 133
sanitize	21
sanitizesort	20
savenumberlist	7
savewrites	25, 361, 362
false	148
true	150, 156
savewrites	25
section	6, 37
section	6
seeautonumberlist	7
shotcuts	23
smallcaps	23
smaller	23
sort	
def	10
standard	10
use	10
sort	10
style	7, 228
style	7
subentrycounter	179, 181
subentrycounter	9
symbols	25
toc	5
true	5
toc	5
translate	21
false	21
translate	21
translator	21
ucmark	9



shortpluralaccess (key) .....	309	sm-short-long-desc (acrstyle) ..	
shotcuts (option) .....	23	.....	203, 331
\showacronymlists .....	232	smallcaps (option) .....	23
\showglocounter .....	229	smaller (option) .....	23
\showglodesc .....	231	\SmallNewAcronymDef ....	222, 341
\showglodescaccess .....	343	sort (key) .....	58
\showglodescplural .....	231	sort (option) .....	10
\showglodescpluralaccess ...	343	style (option) .....	7
\showglofirst .....	229	subentrycounter (option) .....	9
\showglofirstaccess .....	343	\subglossentry .....	183
\showglofirsttpl .....	229	\subitem .....	280
\showglofirstpluralaccess ..	343	sublistdotted (style) .....	251
\showgloflag .....	232	\subsubitem .....	280
\showgloindex .....	232	super (style) .....	267
\showglolevel .....	228	super3col (style) .....	269
\showgloloclist .....	232	super3colborder (style) .....	270
\showglolong .....	232	super3colheader (style) .....	270
\showglolongaccess .....	344	super3colheaderborder (style) .	270
\showglolongpluralaccess ...	344	super4col (style) .....	271
\showglongame .....	231	super4colborder (style) .....	272
\showglongameaccess .....	342	super4colheader (style) .....	271
\showgloparent .....	228	super4colheaderborder (style) .	272
\showgloplural .....	229	superborder (style) .....	268
\showglopluralaccess .....	343	superheader (style) .....	268
\showgloshort .....	232	superheaderborder (style) .....	269
\showgloshortaccess .....	343	superragged (style) .....	274
\showgloshortpluralaccess ..	343	superragged3col (style) .....	276
\showglosort .....	231	superragged3colborder (style) .	277
\showglossaries .....	233	superragged3colheader (style) .	277
\showglossarycounter .....	233	superraggedborder (style) .....	275
\showglossaryentries .....	233	superraggedheader (style) .....	275
\showglossaryin .....	233	superraggedheaderborder (style)	276
\showglossaryout .....	233	superraggedright3colheaderborder	
\showglossarytitle .....	233	(style) .....	277
\showglosymbol .....	231	supertabular (environment) ...	
\showglosymbolaccess .....	343	.....	8, 228, 267–279
\showglosymbolplural .....	231	supertabular package ..	8, 228, 267, 274
\showglosymbolpluralaccess .	343	symbol (key) .....	59
\showglotext .....	229	symbolaccess (key) .....	309
\showglotextaccess .....	343	\symbolname .....	30
\showglotype .....	229	symbolplural (key) .....	59
\showglouserii .....	230	symbolpluralaccess (key) .....	309
\showglouseriii .....	230	symbols (option) .....	25
\showglouseriv .....	230		
\showglouserv .....	230		
\showglouservi .....	230		
sm-short-long (acrstyle) .....	202		

## T

text (key) .....	58
textaccess (key) .....	308
textcase package .....	3
\theequation .....	161

theglossary (environment) ... 5,  
     16, 35, 183, 189, 265, 266, 282–285  
 \theHequation ..... 161  
 theindex (environment) ..... 280  
 toc (option) ..... 5  
 \translate ..... 30  
 translate (option) ..... 21  
 translator package ..... 12–14,  
     26, 28, 30, 31, 164, 344, 353–356  
 tree (style) ..... 281  
 treegroup (style) ..... 282  
 treehypergroup (style) ..... 282  
 treenoname (style) ..... 283  
 treenonamegroup (style) ..... 284  
 treenonamehypergroup (style) .. 284  
 type (key) ..... 59

## U

ucmark (option) ..... 9  
 user1 (key) ..... 60  
 user2 (key) ..... 60

user3 (key) ..... 61  
 user4 (key) ..... 61  
 user5 (key) ..... 61  
 user6 (key) ..... 61

## W

\warn@nomakeglossaries ..... 150  
 \warn@noprintglossary ..... 163  
 \writeist 33, 40, 41, 44, 142, 288, 290

## X

\xcapitalisewords ..... 243  
 \xglsaccsupp ..... 313  
 xindy ..... 357  
 xindy ..... 10, 23, 24, 33,  
     39, 42, 44, 46, 47, 77, 103, 141,  
     142, 144, 160, 166, 186, 234, 289  
 xindy (option) ..... 24  
 xindygloss (option) ..... 24  
 xindynoglsnumbers (option) .... 24  
 \xmakefirsttuc ..... 242  
 xspace package ..... 4, 192