

Documented Code For glossaries v4.02

Nicola L.C. Talbot

Dickimaw Books

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

2013-12-05

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 v4.02: L^AT_EX2e Package to Assist Generating Glossaries”.

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

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

INSTALL Installation instructions.

CHANGES Change log.

README Package summary.

Contents

1 Main Package Code	3
1.1 Package Definition	3
1.2 Package Options	5
1.3 Default values	26
1.4 Xindy	36
1.5 Loops and conditionals	44
1.6 Defining new glossaries	48
1.7 Defining new entries	51
1.8 Resetting and unsetting entry flags	71
1.9 Loading files containing glossary entries	73
1.10 Using glossary entries in the text	73
1.10.1 Links to glossary entries	84
1.10.2 Displaying entry details without adding information to the glossary	135
1.11 Adding an entry to the glossary without generating text	142
1.12 Creating associated files	143
1.13 Writing information to associated files	152
1.14 Glossary Entry Cross-References	157
1.15 Displaying the glossary	159
1.16 Acronyms	176
1.17 Predefined acronym styles	181
1.18 Predefined Glossary Styles	211
1.19 Debugging Commands	212
1.20 Compatibility with version 2.07 and below	217
2 Prefix Support (glossaries-prefix Code)	217
3 Mfirstuc Documented Code	224
4 Glossary Styles	226
4.1 Glossary hyper-navigation definitions (glossary-hypernav package)	226
4.2 In-line Style (glossary-inline.sty)	228
4.3 List Style (glossary-list.sty)	231
4.4 Glossary Styles using longtable (the glossary-long package)	234
4.5 Glossary Styles using longtable (the glossary-longragged package)	240
4.6 Glossary Styles using multicol (glossary-mcols.sty)	245
4.7 Glossary Styles using supertabular environment (glossary-super package)	249
4.8 Glossary Styles using supertabular environment (glossary-superragged package)	256
4.9 Tree Styles (glossary-tree.sty)	262
5 glossaries-compatible-207	270

6 Accessibility Support (glossaries-accsupp Code)	289
6.1 Defining Replacement Text	290
6.2 Accessing Replacement Text	294
6.3 Displaying the Glossary	305
6.4 Acronyms	306
6.5 Debugging Commands	309
7 Multi-Lingual Support	311
7.1 Babel Captions	311
7.2 Polyglossia Captions	317
7.3 Brazilian Dictionary	320
7.4 Danish Dictionary	320
7.5 Dutch Dictionary	321
7.6 English Dictionary	321
7.7 French Dictionary	321
7.8 German Dictionary	321
7.9 Irish Dictionary	322
7.10 Italian Dictionary	322
7.11 Magyar Dictionary	322
7.12 Polish Dictionary	323
7.13 Serbian Dictionary	323
7.14 Spanish Dictionary	323
Glossary	323
Change History	324
Index	339

1 Main Package Code

1.1 Package Definition

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

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries}[2013/12/05 v4.02 (NLCT)]
```

Required packages:

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

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

```
6 \RequirePackage{textcase}
7 \renewcommand*{\mfirstucMakeUppercase}{\MakeTextUppercase}%
```

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

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

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

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

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

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

Check if doc has been loaded.

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

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

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

```
14 {%
```

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

```
16 }%
```

```
17 {%
```

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

```
19 }
```

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

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

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

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

```
22   \@bsphack
```

```
23     \begingroup
```

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

```
25 }
```

```
\glsorg@wrglossary
```

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

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

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

```
29   \endgroup
```

```
30   \@esphack
```

```
31 }
```

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

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

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

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

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

```
37 }
```

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

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

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

`\glsorg@theglossary`

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

`\glsorg@endtheglossary`

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

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

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

End of doc stuff.

```
55 \fi
```

1.2 Package Options

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

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

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

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

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

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

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

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

Determine whether or not to use numbered sections.

`\@@glossarysecstar`

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

`\@@glossaryseclabel`

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

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

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

`numberedsection`

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

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

`\@glossary@default@style`

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

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

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

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

`\@gls@declareoption`

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

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

`\glossaryentrynumbers`

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

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

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

savenumberlist Provide means to store the number list for entries.

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

`\@glo@seeautonumberlist`

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

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

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

```

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

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

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

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

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

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

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

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

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

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

```

```

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

```

nopostdot Boolean option to suppress post description dot

```

126 \define@boolkey{glossaries.sty}[gls]{nopostdot}[true]{}
127 \glsnopostdotfalse

```

nogroupskip Boolean option to suppress vertical space between groups in the pre-defined styles.

```

128 \define@boolkey{glossaries.sty}[gls]{nogroupskip}[true]{}
129 \glsnogroupskipfalse

```

ucmark Boolean option to determine whether or not to use 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

```

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

```

147 \define@choicekey{glossaries.sty}{sort}{standard,def,use}{%
148   \csname @gls@setupsort@#1\endcsname
149 }

```

`\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.

```

150 \newcommand*{\glsprestandardsort}[3]{%
151   \glsdosanitizesort
152 }

```

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

```

153 \newcommand*{\@gls@setupsort@standard}{%

```

Store entry information when it's defined.

```

154   \def\do@glo@storeentry{\@glo@storeentry}%

```

No count register required for standard sort.

```

155   \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.)

```

156   \def\@gls@defsort##1##2{%

```

```

157     \ifx\@glo@sort\@glsdefaultsort

```

```

158       \let\@glo@sort\@glo@name

```

```

159     \fi

```

```

160     \let\glsdosanitizesort\@gls@sanitizesort

```

```

161     \glsprestandardsort{\@glo@sort}{##1}{##2}%

```

```

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

```

```

163   }%

```

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

```

164   \def\@gls@setsort##1{%

```

```

165 }

```

Set standard sort as the default:

```

166 \@gls@setupsort@standard

```

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

```

167 \newcommand*{\glssortnumberfmt}[1]{%

```

```

168   \ifnum#1<100000 0\fi

```

```

169   \ifnum#1<10000 0\fi

```

```

170   \ifnum#1<1000 0\fi

```

```

171   \ifnum#1<100 0\fi

```

```

172 \ifnum#1<10 0\fi
173 \number#1%
174 }

\@gls@setupsort@def Set up the macros for order of definition sorting.
175 \newcommand*{\@gls@setupsort@def}{%
    Store entry information when it's defined.
176 \def\do@glo@storeentry{\@glo@storeentry}%
    Defined count register associated with the glossary.
177 \def\@gls@defsortcount##1{%
178 \expandafter\global
179 \expandafter\newcount\csname glossary@##1@sortcount\endcsname
180 }%
    Increment count register associated with the glossary and use as the sort key.
181 \def\@gls@defsort##1##2{%
182 \expandafter\global\expandafter
183 \advance\csname glossary@##1@sortcount\endcsname by 1\relax
184 \expandafter\protected@xdef\csname glo@##2@sort\endcsname{%
185 \expandafter\glssortnumberfmt
186 {\csname glossary@##1@sortcount\endcsname}}%
187 }%
    Don't need to do anything when the entry is used.
188 \def\@gls@setsort##1{%
189 }

\@gls@setupsort@use Set up the macros for order of use sorting.
190 \newcommand*{\@gls@setupsort@use}{%
    Don't store entry information when it's defined.
191 \let\do@glo@storeentry\@gobble
    Defined count register associated with the glossary.
192 \def\@gls@defsortcount##1{%
193 \expandafter\global
194 \expandafter\newcount\csname glossary@##1@sortcount\endcsname
195 }%
    Initialise the sort key to empty.
196 \def\@gls@defsort##1##2{%
197 \expandafter\gdef\csname glo@##2@sort\endcsname{}%
198 }%
    If the sort key hasn't been set, increment the counter associated with the glossary and set the sort key.
199 \def\@gls@setsort##1{%
    Get the parent, if one exists
200 \edef\@glo@parent{\csname glo@##1@parent\endcsname}%

```

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

```

201 \ifx\@glo@parent\@empty
202 \else
203 \expandafter\@gls@setsort\expandafter{\@glo@parent}%
204 \fi
Set index information for this entry
205 \edef\@glo@type{\csname glo@##1@type\endcsname}%
206 \edef\@gls@tmp{\csname glo@##1@sort\endcsname}%
207 \ifx\@gls@tmp\@empty
208 \expandafter\global\expandafter
209 \advance\csname glossary@\@glo@type @sortcount\endcsname by 1\relax
210 \expandafter\protected@xdef\csname glo@##1@sort\endcsname{%
211 \expandafter\glssortnumberfmt
212 {\csname glossary@\@glo@type @sortcount\endcsname}}%
213 \@glo@storeentry{##1}%
214 \fi
215 }%
216 }

```

`\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.)

```

217 \newcommand*\glsdefmain{%
218 \if@gls@docloaded
219 \newglossary[glg2]{main}{gls2}{glo2}{\glossaryname}%
220 \else
221 \newglossary{main}{gls}{glo}{\glossaryname}%
222 \fi
223 }

```

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`

```

224 \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`

```

225 \newcommand*\acronymtype{\glsdefaulttype}

```

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

```
226 \@gls@declareoption{nomain}{%
227   \let\glsdefaulttype\relax
228   \renewcommand*{\glsdefmain}{}%
229 }
```

`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.

```
230 \define@boolkey{glossaries.sty}[gls]{acronym}[true]{%
231   \ifglsacronym
232     \renewcommand{\@gls@do@acronymsdef}{%
233       \DeclareAcronymList{acronym}%
234       \newglossary[alg]{acronym}{acr}{acn}{\acronymname}%
235       \renewcommand*{\acronymtype}{acronym}%
236     }%
237   \else
238     \let\@gls@do@acronymsdef\relax
239   \fi
240 }
```

`\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.

```
241 \AtBeginDocument{%
242   \ifglsacronym
243     \ifbool{glscompatible-3.07}%
244       {}%
245       {%
246         \providecommand*{\printacronyms}[1][ ]{%
247           \printglossary[type=\acronymtype,#1]}%
248       }%
249   \fi
250 }
```

`@gls@do@acronymsdef` Set default value

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

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

```
252 \@gls@declareoption{acronyms}{%
253   \glsacronymtrue
254   \renewcommand{\@gls@do@acronymsdef}{%
255     \DeclareAcronymList{acronym}%
256     \newglossary[alg]{acronym}{acr}{acn}{\acronymname}%
257     \renewcommand*{\acronymtype}{acronym}%
258   }%
259 }
```

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

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

`\@addtoacronymlists`

```
261 \newcommand*{\@addtoacronymlists}[1]{%
262   \ifx\@glsacronymlists\@empty
263     \protected@xdef\@glsacronymlists{#1}%
264   \else
265     \protected@xdef\@glsacronymlists{\@glsacronymlists,#1}%
266   \fi
267 }
```

`\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.)

```
268 \newcommand*{\DeclareAcronymList}[1]{%
269   \glsIfListOfAcronyms{#1}{\@addtoacronymlists{#1}}%
270 }
```

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

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

```
271 \newcommand{\glsIfListOfAcronyms}[1]{%
272   \edef\@do@glis@islistofacronyms{%
273     \noexpand\@glis@islistofacronyms{#1}{\@glsacronymlists}}%
274   \@do@glis@islistofacronyms
275 }
```

Internal command requires label and list to be expanded:

```
276 \newcommand{\@glis@islistofacronyms}[4]{%
277   \def\@glis@islistofacronyms##1,#1,##2\end@glis@islistofacronyms{%
278     \def\@before{##1}\def\@after{##2}}%
279   \glis@islistofacronyms,#2,#1,\@nil\end@glis@islistofacronyms
280   \ifx\@after\@nnil
```

Not found

```
281   #4%
282   \else
```

Found

```
283   #3%
284   \fi
285 }
```

`if@glsisacronymlist` Convenient boolean.

```

286 \newif\if@glsisacronymlist

```

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

```

287 \newcommand*{\gls@checkisacronymlist}[1]{%
288   \glsIfListOfAcronyms{#1}%
289   {\@glsisacronymlisttrue}{\@glsisacronymlistfalse}%
290 }

```

`\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.)

```

291 \newcommand*{\SetAcronymLists}[1]{%
292   \renewcommand*{\@glsacronymlists}{#1}%
293 }

```

`acronymlists`

```

294 \define@key{glossaries.sty}{acronymlists}{%
295   \DeclareAcronymList{#1}%
296 }

```

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`

```

297 \newcommand{\glscounter}{page}

```

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

```

298 \define@key{glossaries.sty}{counter}{%
299   \renewcommand*{\glscounter}{#1}%
300 }

```

`\@gls@nohyperlist`

```

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

```

`sDeclareNoHyperList`

```

302 \newcommand*{\GlsDeclareNoHyperList}[1]{%
303   \ifdefempty\@gls@nohyperlist
304   {%
305     \renewcommand*{\@gls@nohyperlist}{#1}%
306   }%
307   {%
308     \appto\@gls@nohyperlist{,#1}%
309   }%
310 }

```

nohypertypes

```
311 \define@key{glossaries.sty}{nohypertypes}{%
312   \GlsDeclareNoHyperList{#1}%
313 }
```

\GlossariesWarning Prints a warning message.

```
314 \newcommand*{\GlossariesWarning}[1]{%
315   \PackageWarning{glossaries}{#1}%
316 }
```

sariesWarningNoLine Prints a warning message without the line number.

```
317 \newcommand*{\GlossariesWarningNoLine}[1]{%
318   \PackageWarningNoLine{glossaries}{#1}%
319 }
```

nowarn Define package option to suppress warnings

```
320 \@gls@declareoption{nowarn}{%
321   \renewcommand*{\GlossariesWarning}[1]{}%
322   \renewcommand*{\GlossariesWarningNoLine}[1]{}%
323 }
```

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

```
324 \newcommand*{\@gls@sanitizedesc}{%
325 }
326 %\end{macro}
327 %
328 %\begin{macro}{\glssetexpandfield}
329 %\changes{3.13a}{2013-11-05}{new}
330 %\begin{definition}
331 %\cs{glssetexpandfield}\marg{field}
332 %\end{definition}
333 % Sets field to always expand.
334 %   \begin{macrocode}
335 \newcommand*{\glssetexpandfield}[1]{%
336   \csdef{gls@assign@#1@field}##1##2{%
337     \@gls@expand@field{##1}{#1}{##2}%
338   }%
339 }
```

\glssetnoexpandfield

\glssetnoexpandfield{<field>}

Sets field to never expand.

```

340 \newcommand*{\glsetnoexpandfield}[1]{%
341   \csdef{gls@assign@#1@field}##1##2{%
342     \@@gls@noexpand@field{##1}{#1}{##2}%
343   }%
344 }

```

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

```
345 \glsetexpandfield{type}
```

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

```
346 \glsetnoexpandfield{desc}
```

gn@descplural@field

```
347 \glsetnoexpandfield{descplural}
```

\@gls@sanitizename

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

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

```
349 \glsetnoexpandfield{name}
```

@gls@sanitizesymbol

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

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

```
351 \glsetnoexpandfield{symbol}
```

@symbolplural@field

```
352 \glsetnoexpandfield{symbolplural}
```

Sanitizing stuff:

\@gls@sanitizesort

```

353 \newcommand*{\@gls@sanitizesort}{%
354   \ifglssanitizesort
355     \@onelevel@sanitize\@glo@sort
356   \else
357     \fi
358 }

```

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.

```

359 \define@boolkey[gls]{sanitize}{description}[true]{%
360   \GlossariesWarning{sanitize={description} package option deprecated}%
361   \ifgls@sanitize@description
362     \glsetnoexpandfield{desc}%
363     \glsetnoexpandfield{descplural}%

```

```

364 \else
365   \glssetexpandfield{desc}%
366   \glssetexpandfield{descplural}%
367 \fi
368 }

369 \define@boolkey[gls]{sanitize}{name}[true]{%
370   \GlossariesWarning{sanitize={name} package option
371 deprecated}%
372   \ifgls@sanitize@name
373     \glssetnoexpandfield{name}%
374   \else
375     \glssetexpandfield{name}%
376   \fi
377 }

378 \define@boolkey[gls]{sanitize}{symbol}[true]{%
379   \GlossariesWarning{sanitize={symbol} package option
380 deprecated}%
381   \ifgls@sanitize@symbol
382     \glssetnoexpandfield{symbol}%
383     \glssetnoexpandfield{symbolplural}%
384   \else
385     \glssetexpandfield{symbol}%
386     \glssetexpandfield{symbolplural}%
387   \fi
388 }

```

sanitizesort

```

389 \define@boolkey{glossaries.sty}[gls]{sanitizesort}[true]{%
390   \ifglssanitizesort
391     \glssetnoexpandfield{sortvalue}%
392   \else
393     \glssetexpandfield{sortvalue}%
394   \fi
395 }

  Default setting:

396 \glssanitizesorttrue
397 \glssetnoexpandfield{sortvalue}%

398 \define@choicekey[gls]{sanitize}{sort}{true,false}[true]{%
399   \setbool{glssanitizesort}{#1}%
400   \ifglssanitizesort
401     \glssetnoexpandfield{sortvalue}%
402   \else
403     \glssetexpandfield{sortvalue}%
404   \fi
405   \GlossariesWarning{sanitize={sort} package option
406     deprecated. Use sanitizesort instead}%
407 }

```

```

sanitize
408 \define@key{glossaries.sty}{sanitize}[description=true,symbol=true,
409 name=true]{%
410   \ifthenelse{\equal{#1}{none}}{%
411     {%
412       \GlossariesWarning{sanitize package option deprecated}%
413     }%
414     {%
415       \setkeys[glS]{sanitize}{#1}%
416     }%
417 }

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

ls@nottranslatorhook
419 \newcommand*{\@glS@nottranslatorhook{}}

nottranslate Provide a synonym for translate=false that can be passed via the document
class.
420 \@glS@declareoption{nottranslate}{%
421   \glstranslatefalse
422   \let\@glS@nottranslatorhook\relax
423 }

translate Define translate option. If false don't set up multi-lingual support.
424 \define@choicekey{glossaries.sty}{translate}[\val\nr]%
425 {true,false,babel}[true]%
426 {%
427   \ifcase\nr\relax
428     \glstranslatetrue
429   \or
430     \glstranslatefalse
431     \let\@glS@nottranslatorhook\relax
432   \or
433     \glstranslatefalse
434     \def\@glS@nottranslatorhook{\RequirePackage{glossaries-babel}}%
435   \fi
436 }

Set the default value:
437 \glstranslatefalse
438 \@ifpackageloaded{translator}%
439   {\glstranslatetrue}%
440   {%
441     \@ifpackageloaded{polyglossia}%
442       {\glstranslatetrue}%

```

```

443      {%
444      \@ifpackageloaded{babel}{\glstranslatetrue}{}}%
445      }%
446 }

```

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

```

447 \define@boolkey{glossaries.sty}[gls]{indexonlyfirst}[true]{}
448 \glsindexonlyfirstfalse

```

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

```

449 \define@boolkey{glossaries.sty}[gls]{hyperfirst}[true]{}
450 \glshyperfirsttrue

```

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

```

451 \newcommand*{\@gls@setacrstyle}{}

```

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

```

452 \define@boolkey{glossaries.sty}[glsacr]{footnote}[true]{}%
453 \ifbool{glsacrdescription}%
454 {}%
455 {%
456 \renewcommand*{\@gls@sanitizedesc}{}%
457 }%
458 \renewcommand*{\@gls@setacrstyle}{\SetAcronymStyle}%
459 }

```

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

```

460 \define@boolkey{glossaries.sty}[glsacr]{description}[true]{}%
461 \renewcommand*{\@gls@sanitizesymbol}{}%
462 \renewcommand*{\@gls@setacrstyle}{\SetAcronymStyle}%
463 }

```

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

```

464 \define@boolkey{glossaries.sty}[glsacr]{smallcaps}[true]{}%
465 \renewcommand*{\@gls@sanitizesymbol}{}%
466 \renewcommand*{\@gls@setacrstyle}{\SetAcronymStyle}%
467 }

```

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

```

468 \define@boolkey{glossaries.sty}[glsacr]{smaller}[true]{}%
469 \renewcommand*{\@gls@sanitizesymbol}{}%
470 \renewcommand*{\@gls@setacrstyle}{\SetAcronymStyle}%
471 }

```

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

```

472 \define@boolkey{glossaries.sty}[glsacr]{dua}[true]{%
473   \renewcommand*{\@gls@sanitizesymbol}{}%
474   \renewcommand*{\@gls@setacrstyle}{\SetAcronymStyle}%
475 }

```

shortcuts Define acronym shortcuts.

```

476 \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.

```

477 \newcommand*{\glsorder}{word}

```

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

```

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

```

order

```

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

```

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

```

481 \newif\ifglxindy

```

The default is makeindex:

```

482 \glxindyfalse

```

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

```

483 \@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.

```

484 \define@boolkey[gls]{xindy}{glsnumbers}[true]{}
485 \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.)

```

486 \def\@xdy@main@language{\language}%

```

Define key to set the language

```

487 \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.

```
488 \ifcsundef{inputencodingname}{%  
489   \def\gls@codepage{}}{%  
490   \def\gls@codepage{\inputencodingname}  
491 }
```

Define a key to set the code page.

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

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

```
493 \define@key{glossaries.sty}{xindy}[]{%  
494   \glsxindytrue  
495   \setkeys[gls]{xindy}{#1}%  
496 }
```

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

```
497 \@gls@declareoption{xindygloss}{%  
498   \glsxindytrue  
499 }
```

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

```
500 \@gls@declareoption{xindynoglsnumbers}{%  
501   \glsxindytrue  
502   \gls@xindy@glsnumbersfalse  
503 }
```

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

```
504 \define@boolkey{glossaries.sty}[gls]{savewrites}[true]{%  
505   \ifglssavewrites  
506     \renewcommand*{\glswritefiles}{\@glswritefiles}%  
507   \else  
508     \let\glswritefiles\relax  
509   \fi  
510 }
```

Set default:

```
511 \glssavewritesfalse  
512 \let\glswritefiles\relax
```

`compatible-3.07`

```
513 \define@boolkey{glossaries.sty}[gls]{compatible-3.07}[true]{%  
514 \boolfalse{glscompatible-3.07}}
```

`compatible-2.07`

```
515 \define@boolkey{glossaries.sty}[gls]{compatible-2.07}[true]{%
```

Also set 3.07 compatibility if this option is set.

```
516 \ifbool{glscompatible-2.07}%  
517 {%  
518 \booltrue{glscompatible-3.07}%  
519 }%  
520 }%  
521 }  
522 \boolfalse{glscompatible-2.07}
```

symbols Create a “symbols” glossary type

```
523 \@gls@declareoption{symbols}{%  
524 \let\@gls@do@symbolsdef\@gls@symbolsdef  
525 }
```

Default is not to define the symbols glossary:

```
526 \newcommand*{\@gls@do@symbolsdef}{}%
```

\@gls@symbolsdef

```
527 \newcommand*{\@gls@symbolsdef}{%  
528 \newglossary[slg]{symbols}{sls}{slo}{\glssymbolsgroupname}%  
529 \newcommand*{\printsymbols}[1] [] {\printglossary[type=symbols,##1]}%  
530 }%
```

numbers Create a “symbols” glossary type

```
531 \@gls@declareoption{numbers}{%  
532 \let\@gls@do@numbersdef\@gls@numbersdef  
533 }
```

Default is not to define the numbers glossary:

```
534 \newcommand*{\@gls@do@numbersdef}{}%
```

\@gls@numbersdef

```
535 \newcommand*{\@gls@numbersdef}{%  
536 \newglossary[nlg]{numbers}{nls}{nlo}{\glsnumbersgroupname}%  
537 \newcommand*{\printnumbers}[1] [] {\printglossary[type=numbers,##1]}%  
538 }%
```

index Create an “index” glossary type

```
539 \@gls@declareoption{index}{%  
540 \let\@gls@do@indexdef\@gls@indexdef  
541 }
```

Default is not to define index glossary:

```
542 \newcommand*{\@gls@do@indexdef}{}%
```

\@gls@indexdef

```
543 \newcommand*{\@gls@indexdef}{%  
544 \newglossary[ilg]{index}{ind}{idx}{\indexname}%  
545 \newcommand*{\printindex}[1] [] {\printglossary[type=index,##1]}%
```

```

546 \newcommand*{\newterm}[2][]{%
547   \newglossaryentry{##2}%
548   {type={index},name={##2},description={\nopostdesc},##1}}
549 }%

```

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

```

550 \for\CurrentOption :=\@declaredoptions\do{%
551   \ifx\CurrentOption\@empty
552   \else
553     \@expandtwoargs
554     \in@ {,\CurrentOption ,}{,\@classoptionslist,\@curroptions,}%
555     \ifin@
556     \@use@option
557     \expandafter \let\csname ds@\CurrentOption\endcsname\@empty
558     \fi
559   \fi
560 }

```

Now process options passed to the package:

```
561 \ProcessOptionsX
```

Load backward compatibility stuff:

```
562 \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:

```

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

```

Now define `\setupglossaries`:

```

566 \newcommand*{\setupglossaries}[1]{%
567   \renewcommand*{\@gls@setacrstyle}{}%
568   \ifglsacrshortcuts
569     \def\@gls@setupshortcuts{\glsacrshortcutstrue}%
570   \else
571     \def\@gls@setupshortcuts{%
572       \ifglsacrshortcuts
573         \DefineAcronymSynonyms
574       \fi
575     }%
576   \fi
577   \glsacrshortcutsfalse
578   \let\@gls@do@numbersdef\relax
579   \let\@gls@do@symbolssdef\relax
580   \let\@gls@do@indexdef\relax
581   \let\@gls@do@acronymsdef\relax
582   \setkeys{glossaries.sty}{#1}%
583   \@gls@setacrstyle

```

```

584 \@gls@setupshortcuts
585 \@gls@do@acronymsdef
586 \@gls@do@numbersdef
587 \@gls@do@symbolssdef
588 \@gls@do@indexdef
589 }

```

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

```

590 \ifglstranslate
591 \@ifpackageloaded{polyglossia}%
592 {%
    polyglossia fakes babel so need to check for polyglossia first.
593 }%
594 {%
    \@ifpackageloaded{babel}%
595     {%
596         \IfFileExists{translator.sty}%
597         {%
598             \RequirePackage{translator}%
599         }%
600     }%
601 }%
602 }%
603 {}
604 }
605 \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}).

```

606 \ifthenelse{\equal{\glscounter}{section}}{%
607 {%
608     \ifcsundef{chapter}{}%
609     {%
610         \let\@gls@old@chapter\@chapter
611         \def\@chapter[#1]#2{\@gls@old@chapter[#1]{#2}%
612             \ifcsundef{hyperdef}{\hyperdef{section}{\thesection}}}%
613     }%
614 }%
615 {}

```

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

```
616 \newcommand*{\@gls@onlypremakeg}{}
```

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

```
617 \newcommand*{\@onlypremakeg}[1]{%
618   \ifx\@gls@onlypremakeg\@empty
619     \def\@gls@onlypremakeg{#1}%
620   \else
621     \expandafter\toks@\expandafter{\@gls@onlypremakeg}%
622     \edef\@gls@onlypremakeg{\the\toks@,\noexpand#1}%
623   \fi
624 }
```

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

```
625 \newcommand*{\@disable@onlypremakeg}{%
626   \@for\@thiscs:=\@gls@onlypremakeg\do{%
627     \expandafter\@disable@premakecs\@thiscs%
628   }}
```

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

```
629 \newcommand*{\@disable@premakecs}[1]{%
630   \def#1{\PackageError{glossaries}{\string#1\space may only be
631     used before \string\makeglossaries}{You can't use
632     \string#1\space after \string\makeglossaries}}%
633 }
```

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`

```
634 \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`

```
635 \providecommand*{\acronymname}{Acronyms}
```

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

```
636 \newcommand*{\glssettoctitle}[1]{%
637   \def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}
```

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
638 \providecommand*\entryname}{Notation}

\descriptionname
639 \providecommand*\descriptionname}{Description}

\symbolname
640 \providecommand*\symbolname}{Symbol}

\pagelistname
641 \providecommand*\pagelistname}{Page List}

Labels for makeindex's symbol and number groups:
glsymbolsgroupname
642 \providecommand*\glsymbolsgroupname}{Symbols}
glsnumbersgroupname
643 \providecommand*\glsnumbersgroupname}{Numbers}

\glspluralsuffix The default plural is formed by appending \glspluralsuffix to the singular
form.
644 \newcommand*\glspluralsuffix}{s}

\seename
645 \providecommand*\seename}{see}

\andname
646 \providecommand*\andname}{\&}

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

dglossarytocaptions If using , \glossaryname should be defined in terms of \translate, but if ba-
bel is also loaded, it will redefine \glossaryname whenever the language is set,
so override it. (Don't use \addto as doesn't define it.)
647 \newcommand*\addglossarytocaptions}[1]{%
648   \ifcsundef{captions#1}{}%
649   {%
650     \expandafter\let\expandafter\@gls@tmp\csname captions#1\endcsname
651     \expandafter\toks@\expandafter{\@gls@tmp
652       \renewcommand*\glossaryname}{\translate{Glossary}}}%
653   }%
654   \expandafter\edef\csname captions#1\endcsname{\the\toks@}%
655 }%
656 }

```

657 \ifglstranslate

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

```
658 \@ifpackageloaded{translator}{%
659   \usedictionary{glossaries-dictionary}%
660   \addglossarytocaptions{portuges}%
661   \addglossarytocaptions{portuguese}%
662   \addglossarytocaptions{brazil}%
663   \addglossarytocaptions{brazilian}%
664   \addglossarytocaptions{danish}%
665   \addglossarytocaptions{dutch}%
666   \addglossarytocaptions{afrikaans}%
667   \addglossarytocaptions{english}%
668   \addglossarytocaptions{UKenglish}%
669   \addglossarytocaptions{USenglish}%
670   \addglossarytocaptions{american}%
671   \addglossarytocaptions{australian}%
672   \addglossarytocaptions{british}%
673   \addglossarytocaptions{canadian}%
674   \addglossarytocaptions{newzealand}%
675   \addglossarytocaptions{french}%
676   \addglossarytocaptions{frenchb}%
677   \addglossarytocaptions{francais}%
678   \addglossarytocaptions{acadian}%
679   \addglossarytocaptions{canadien}%
680   \addglossarytocaptions{german}%
681   \addglossarytocaptions{germanb}%
682   \addglossarytocaptions{austrian}%
683   \addglossarytocaptions{naustrian}%
684   \addglossarytocaptions{ngerman}%
685   \addglossarytocaptions{irish}%
686   \addglossarytocaptions{italian}%
687   \addglossarytocaptions{magyar}%
688   \addglossarytocaptions{hungarian}%
689   \addglossarytocaptions{polish}%
690   \addglossarytocaptions{spanish}%
691   \renewcommand*{\glstttitle}[1]{%
692     \ifthenelse{\equal{#1}{main}}{%
693       \translatelet{\glossarytoctitle}{Glossary}}{%
694       \ifthenelse{\equal{#1}{acronym}}{%
695         \translatelet{\glossarytoctitle}{Acronyms}}{%
696         \def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}}%
697   \renewcommand*{\glossaryname}{\translate{Glossary}}%
698   \renewcommand*{\acronymname}{\translate{Acronyms}}%
699   \renewcommand*{\entryname}{\translate{Notation (glossaries)}}%
700   \renewcommand*{\descriptionname}{%
701     \translate{Description (glossaries)}}%
702   \renewcommand*{\symbolname}{\translate{Symbol (glossaries)}}%
703   \renewcommand*{\pagelistname}{%
704     \translate{Page List (glossaries)}}%
```

```

705 \renewcommand*{\glssymbolsgroupname}{%
706 \translate{Symbols (glossaries)}}%
707 \renewcommand*{\glsnumbersgroupname}{%
708 \translate{Numbers (glossaries)}}%
709 }{%

710 \@ifpackageloaded{polyglossia}%
711 {\RequirePackage{glossaries-polyglossia}}%
712 {%
713 \@ifpackageloaded{babel}{%
714 \RequirePackage{glossaries-babel}}}%
715 }}
716 \else

717 \@gls@nottranslatorhook
718 \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.

```
719 \DeclareRobustCommand*{\nopostdesc}{}

```

`\@nopostdesc` Suppress next description terminator.

```

720 \newcommand*{\@nopostdesc}{%
721 \let\org@glspostdescription\glspostdescription
722 \def\glspostdescription{%
723 \let\glspostdescription\org@glspostdescription}%
724 }

```

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

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

```

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

```
726 \newcommand{\glspar}{\par}

```

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

```

727 \ifglxindy
728 \newcommand{\setStyleFile}[1]{%
729 \renewcommand{\istfilename}{#1.xdy}}
730 \else
731 \newcommand{\setStyleFile}[1]{%
732 \renewcommand{\istfilename}{#1.ist}}
733 \fi

```

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

```
734 \@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 re-defining 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`

```
735 \ifglxindy
736   \def\istfilename{\jobname.xdy}
737 \else
738   \def\istfilename{\jobname.ist}
739 \fi
```

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`

```
740 \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`

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

`\glsSetCompositor` Sets the compositor.

```
742 \newcommand*\glsSetCompositor[1]{%
743   \renewcommand*\glscompositor{#1}}
Only use before \makeglossaries
744 \@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`.

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

`\sSetAlphaCompositor` Sets the alpha compositor.

```
746 \ifglxindy
747   \newcommand*\glsSetAlphaCompositor[1]{%
748     \renewcommand*\@glsAlphacompositor{#1}}
749 \else
750   \newcommand*\glsSetAlphaCompositor[1]{%
751     \glsnoxindywarning\glsSetAlphaCompositor}
752 \fi
```

Can only be used before `\makeglossaries`
753 `\@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.
754 `\newcommand*\{gls@suffixF}\{ }`

`\glsSetSuffixF` Sets the suffix to use for a two page list.
755 `\newcommand*\{glsSetSuffixF}[1]{%`
756 `\renewcommand*\{gls@suffixF}\{#1}}`
Only has an effect when used before `\makeglossaries`
757 `\@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.
758 `\newcommand*\{gls@suffixFF}\{ }`

`\glsSetSuffixFF` Sets the suffix to use for a three page list.
759 `\newcommand*\{glsSetSuffixFF}[1]{%`
760 `\renewcommand*\{gls@suffixFF}\{#1}%`
761 `}`

`\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".
762 `\ifcsundef{hyperlink}%`
763 `{%`
764 `\newcommand*\{glsnumberformat}[1]{#1}%`
765 `}%`
766 `{%`
767 `\newcommand*\{glsnumberformat}[1]{\glshypernumber{#1}}%`
768 `}`

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`
769 `\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`
770 `\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`

```
771 \newcommand*\glossarypreamble{%
772   \csuse{@glossarypreamble@\currentglossary}%
773 }
```

`\setglossarypreamble`

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

Code provided by Michael Pock.

```
774 \newcommand\setglossarypreamble[2][\glsdefaulttype]{%
775   \ifglossaryexists{#1}{%
776     \csgdef{@glossarypreamble@#1}{#2}%
777   }{%
778     \GlossariesWarning{%
779       Glossary ‘#1’ is not defined%
780     }%
781   }%
782 }
```

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`

```
783 \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`.

```

784 \newcommand*{\glossarysection}[2][\@gls@title]{%
785   \def\@gls@title{#2}%
786   \ifcsundef{phantomsection}%
787   {%
788     \@glossarysection{#1}{#2}%
789   }%
790   {%
791     \@pglossarysection{#1}{#2}%
792   }%

793   \glsglossarymark{\glossarytoctitle}%
794 }

```

`\glsglossarymark` Sets the header mark for the glossary. Takes the glossary short (TOC) title as the argument.

```

795 \ifcsundef{glossarymark}%
796 {%
797   \newcommand{\glsglossarymark}[1]{\glossarymark{#1}}
798 }%
799 {%
800   \@ifclassloaded{memoir}
801   {%
802     \newcommand{\glsglossarymark}[1]{%
803       \ifglsucmark
804         \markboth{\memUHead{#1}}{\memUHead{#1}}%
805       \else
806         \markboth{#1}{#1}%
807       \fi
808     }
809   }%
810   {%
811     \newcommand{\glsglossarymark}[1]{%
812       \ifglsucmark
813         \@mkboth{\mfirstucMakeUppercase{#1}}{\mfirstucMakeUppercase{#1}}%
814       \else
815         \@mkboth{#1}{#1}%
816       \fi
817     }
818   }
819 }

```

`\glossarymark` Provided for backward compatibility:

```

820 \providecommand{\glossarymark}[1]{%
821   \ifglsucmark
822     \@mkboth{\mfirstucMakeUppercase{#1}}{\mfirstucMakeUppercase{#1}}%
823   \else
824     \@mkboth{#1}{#1}%
825   \fi
826 }

```

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`

```
827 \newcommand*\setglossarysection}[1]{%
828 \setkeys{glossaries.sty}{section=#1}}
```

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

`\@glossarysection`

```
829 \newcommand*\@glossarysection}[2]{%
830 \ifdefempty\@@glossarysecstar
831 {%
832 \csname\@@glossarysec\endcsname{#2}%
833 }%
834 {%
835 \csname\@@glossarysec\endcsname*{#2}%
836 \@gls@toc{#1}{\@@glossarysec}%
837 }%
```

Do automatic labelling if required

```
838 \@@glossaryseclabel
839 }
```

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`

```
840 \newcommand*\@pglossarysection}[2]{%
841 \glsclearpage
842 \phantomsection
843 \ifdefempty\@@glossarysecstar
844 {%
845 \csname\@@glossarysec\endcsname{#2}%
846 }%
847 {%
848 \@gls@toc{#1}{\@@glossarysec}%
849 \csname\@@glossarysec\endcsname*{#2}%
850 }%
```

Do automatic labelling if required

```
851 \@@glossaryseclabel
852 }
```

`\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.

```

853 \newcommand*{\gls@doclearpage}{%
854   \ifthenelse{\equal{\@glossarysec}{chapter}}{%
855     {%
856       \ifcsundef{cleardoublepage}%
857       {%
858         \clearpage
859       }%
860     }%
861     \ifcsdef{if@openright}%
862     {%
863       \if@openright
864         \cleardoublepage
865       \else
866         \clearpage
867       \fi
868     }%
869     {%
870       \cleardoublepage
871     }%
872   }%
873 }%
874 {}%
875 }

```

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

```

876 \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`

```

877 \newcommand*{\@gls@toc}[2]{%
878   \ifglstoc
879     \ifglslnumberline
880       \addcontentsline{toc}{#2}{\numberline{#1}}%
881     \else
882       \addcontentsline{toc}{#2}{#1}%
883     \fi
884   \fi
885 }

```

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

```
886 \newcommand*{\glsnoxindywarning}[1]{%
887   \GlossariesWarning{Not in xindy mode --- ignoring \string#1}%
888 }
```

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

```
889 \ifglsxindy
890   \edef\@xdyattributes{\string"default\string"}%
891 \fi
```

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

```
892 \ifglsxindy
893   \edef\@xdyattributelist{}%
894 \fi
```

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

```
895 \ifglsxindy
896   \def\@xdylocref{}
897 \fi
```

`\@gls@ifinlist`

```
898 \newcommand*{\@gls@ifinlist}[4]{%
899   \def\@do@ifinlist##1,#1,##2\end@do@ifinlist{%
900     \def\@gls@listsuffix{##2}%
901     \ifx\@gls@listsuffix\@empty
902       #4%
903     \else
904       #3%
905     \fi
906   }%
907   \@do@ifinlist,#2,#1,\end@do@ifinlist
908 }
```

`\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.

```
909 \ifglsxindy
910   \newcommand*{\@xdycounters}{\glscounter}
911   \newcommand*\GlsAddXdyCounters[1]{%
912     \@for\@gls@ctr:=#1\do{%
```

Check if already in list before adding.

```

913     \edef\@do@addcounter{%
914         \noexpand\@gls@ifinlist{\@gls@ctr}{\@xdycounters}{}%
915         {%
916             \noexpand\edef\noexpand\@xdycounters{\@xdycounters,%
917                 \noexpand\@gls@ctr}%
918         }%
919     }%
920     \@do@addcounter
921 }
922 }

```

Only has an effect before `\writeist`:

```

923 \@onlypremakeg\GlsAddXdyCounters
924 \else
925 \newcommand*\GlsAddXdyCounters[1]{%
926     \glsnoxindywarning\GlsAddXdyAttribute
927 }
928 \fi

```

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

```

929 \newcommand*\@disabled@glssaddxdycounters{%
930     \PackageError{glossaries}{\string\GlsAddXdyCounters\space
931         can't be used after \string\GlsAddXdyAttribute}{Move all
932         occurrences of \string\GlsAddXdyCounters\space before the first
933         instance of \string\GlsAddXdyAttribute}%
934 }

```

`\GlsAddXdyAttribute` Adds an attribute.

```

935 \ifglsxindy

```

First define internal command that adds an attribute for a given counter (2nd argument is the counter):

```

936 \newcommand*\@glssaddxdyattribute[2]{%

```

Add to xindy attribute list

```

937     \edef\@xdyattributes{\@xdyattributes ^^J \string"#1\string" ^^J
938         \string"#2#1\string"}%

```

Add to xindy markup location.

```

939     \expandafter\toks@\expandafter{\@xdylocref}%
940     \edef\@xdylocref{\the\toks@ ^^J%
941         (markup-locref
942         :open \string"\string~n%
943         \expandafter\string\csname glsX#2X#1\endcsname
944         \string" ^^J
945         :close \string"\string" ^^J
946         :attr \string"#2#1\string")}%

```

Define associated attribute command `\glsX<counter>X<attribute>{\<Hprefix>}{\<n>}`

```

947     \expandafter\gdef\csname glsX#2X#1\endcsname##1##2{%

```

```

948     \setentrycounter[##1]{#2}\csname #1\endcsname{##2}%
949   }%
950 }

```

High-level command:

```

951 \newcommand*\GlsAddXdyAttribute[1]{%

```

Add to comma-separated attribute list

```

952   \ifx\@xdyattributelist\@empty
953     \edef\@xdyattributelist{#1}%
954   \else
955     \edef\@xdyattributelist{\@xdyattributelist,#1}%
956   \fi

```

Iterate through all specified counters and add counter-dependent attributes:

```

957   \@for\@this@counter:=\@xdycounters\do{%
958     \protected@edef\gls@do@addxdyattribute{%
959       \noexpand\@glsaddxdyattribute{#1}{\@this@counter}%
960     }
961     \gls@do@addxdyattribute
962   }%

```

All occurrences of `\GlsAddXdyCounters` must be used before this command

```

963   \let\GlsAddXdyCounters\@disabled@glsaddxdycounters
964 }

```

Only has an effect before `\writeist`:

```

965 \@onlypremakeg\GlsAddXdyAttribute
966 \else
967   \newcommand*\GlsAddXdyAttribute[1]{%
968     \glsnoxindywarning\GlsAddXdyAttribute}
969 \fi

```

redefinedattributes Add known attributes for all defined counters

```

970 \ifglxsindy
971 \newcommand*{\@gls@addpredefinedattributes}{%
972   \GlsAddXdyAttribute{glsnumberformat}
973   \GlsAddXdyAttribute{textrm}
974   \GlsAddXdyAttribute{textsf}
975   \GlsAddXdyAttribute{texttt}
976   \GlsAddXdyAttribute{textbf}
977   \GlsAddXdyAttribute{textmd}
978   \GlsAddXdyAttribute{textit}
979   \GlsAddXdyAttribute{textup}
980   \GlsAddXdyAttribute{textsl}
981   \GlsAddXdyAttribute{textsc}
982   \GlsAddXdyAttribute{emph}
983   \GlsAddXdyAttribute{glshypernumber}
984   \GlsAddXdyAttribute{hyperrrm}
985   \GlsAddXdyAttribute{hypersf}
986   \GlsAddXdyAttribute{hypertt}

```

```

987 \GlsAddXdyAttribute{hyperbf}
988 \GlsAddXdyAttribute{hypermd}
989 \GlsAddXdyAttribute{hyperit}
990 \GlsAddXdyAttribute{hyperup}
991 \GlsAddXdyAttribute{hypersl}
992 \GlsAddXdyAttribute{hypersc}
993 \GlsAddXdyAttribute{hyperemph}
994 }
995 \else
996 \let\@gls@addpredefinedattributes\relax
997 \fi

```

`\@xdyuseralphabets` List of additional alphabets

```

998 \def\@xdyuseralphabets{}

```

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

```

999 \ifglsxindy
1000 \newcommand*{\GlsAddXdyAlphabet}[2]{%
1001 \edef\@xdyuseralphabets{%
1002 \@xdyuseralphabets ^^J
1003 (define-alphabet "#1" (#2))}}
1004 \else
1005 \newcommand*{\GlsAddXdyAlphabet}[2]{%
1006 \glsnoxywarning\GlsAddXdyAlphabet}
1007 \fi

```

This code is only required for xindy:

```

1008 \ifglsxindy

```

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

```

1009 \newcommand*{\@gls@xdy@locationlist}{%
1010 roman-page-numbers,%
1011 Roman-page-numbers,%
1012 arabic-page-numbers,%
1013 alpha-page-numbers,%
1014 Alpha-page-numbers,%
1015 Appendix-page-numbers,%
1016 arabic-section-numbers%
1017 }

```

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.

```

1018 \protected\edef\@gls@roman{\@roman{0}\string"

```

```

1019     \string"roman-numbers-lowercase\string" :sep \string"}}%
1020 \@onelevel@sanitize\@gls@roman
1021 \edef\@tmp{\string" \string"roman-numbers-lowercase\string"
1022     :sep \string"}%
1023 \@onelevel@sanitize\@tmp
1024 \ifx\@tmp\@gls@roman
1025     \expandafter
1026     \edef\csname @gls@xdy@Lclass@roman-page-numbers\endcsname{%
1027         \string"roman-numbers-lowercase\string"%
1028     }%
1029 \else
1030     \expandafter
1031     \edef\csname @gls@xdy@Lclass@roman-page-numbers\endcsname{
1032         :sep \string"\@gls@roman\string"%
1033     }%
1034 \fi

```

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

```

1035 \expandafter\def\csname @gls@xdy@Lclass@Roman-page-numbers\endcsname{%
1036     \string"roman-numbers-uppercase\string"%
1037 }%

```

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

```

1038 \expandafter\def\csname @gls@xdy@Lclass@arabic-page-numbers\endcsname{%
1039     \string"arabic-numbers\string"%
1040 }%

```

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

```

1041 \expandafter\def\csname @gls@xdy@Lclass@alpha-page-numbers\endcsname{%
1042     \string"alpha\string"%
1043 }%

```

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

```

1044 \expandafter\def\csname @gls@xdy@Lclass@Alpha-page-numbers\endcsname{%
1045     \string"ALPHA\string"%
1046 }%

```

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

```

1047 \expandafter\def\csname @gls@xdy@Lclass@Appendix-page-numbers\endcsname{%
1048     \string"ALPHA\string"
1049     :sep \string"\@glsAlphacompositor\string"
1050     \string"arabic-numbers\string"%
1051 }

```

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

```

1052 \expandafter\def\csname @gls@xdy@Lclass@arabic-section-numbers\endcsname{%

```

```

1053   \string"arabic-numbers\string"
1054   :sep \string"\glscompositor\string"
1055   \string"arabic-numbers\string"%
1056 }%

```

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

```

1057 \def\xdyuserlocationdefs{}

```

`\xdyuserlocationnames` List of additional user location names

```

1058 \def\xdyuserlocationnames{}

```

End of xindy-only block:

```

1059 \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.)

```

1060 \ifglsxindy
1061   \newcommand*\GlsAddXdyLocation[3][{}]{%
1062     \def\@gls@tmp{#1}%
1063     \ifx\@gls@tmp\@empty
1064       \edef\xdyuserlocationdefs{%
1065         \xdyuserlocationdefs ^^J%
1066         (define-location-class \string"#2\string"^^J\space\space
1067         \space(:sep \string"{}\glsopenbrace\string" #3
1068         :sep \string"\glsclosebrace\string"))
1069       }%
1070     \else
1071       \edef\xdyuserlocationdefs{%
1072         \xdyuserlocationdefs ^^J%
1073         (define-location-class \string"#2\string"^^J\space\space
1074         \space(:sep "\glsopenbrace"
1075         #1
1076         :sep "\glsclosebrace\glsopenbrace" #3
1077         :sep "\glsclosebrace"))
1078       }%
1079     \fi
1080     \edef\xdyuserlocationnames{%
1081       \xdyuserlocationnames^^J\space\space\space
1082       \string"#1\string"}%
1083   }

```

Only has an effect before `\writeist`:

```

1084 \@onlypremake\GlsAddXdyLocation
1085 \else
1086   \newcommand*\GlsAddXdyLocation[2]{%
1087     \glsnnoxindywarning\GlsAddXdyLocation}
1088 \fi

```

ylocationclassorder Define location class order

```
1089 \ifglxindy
1090   \edef\@xdylocationclassorder{^^J\space\space\space
1091     \string"roman-page-numbers\string"^^J\space\space\space
1092     \string"arabic-page-numbers\string"^^J\space\space\space
1093     \string"arabic-section-numbers\string"^^J\space\space\space
1094     \string"alpha-page-numbers\string"^^J\space\space\space
1095     \string"Roman-page-numbers\string"^^J\space\space\space
1096     \string"Alpha-page-numbers\string"^^J\space\space\space
1097     \string"Appendix-page-numbers\string"
1098     \@xdyuserlocationnames^^J\space\space\space
1099     \string"see\string"
1100   }
1101 \fi
```

Change the location order.

yLocationClassOrder

```
1102 \ifglxindy
1103   \newcommand*\GlsSetXdyLocationClassOrder[1]{%
1104     \def\@xdylocationclassorder{#1}}
1105 \else
1106   \newcommand*\GlsSetXdyLocationClassOrder[1]{%
1107     \glsnxindywarning\GlsSetXdyLocationClassOrder}
1108 \fi
```

\@xdysortrules Define sort rules

```
1109 \ifglxindy
1110   \def\@xdysortrules{}
1111 \fi
```

\GlsAddSortRule Add a sort rule

```
1112 \ifglxindy
1113   \newcommand*\GlsAddSortRule[2]{%
1114     \expandafter\toks@\expandafter{\@xdysortrules}%
1115     \protected@edef\@xdysortrules{\the\toks@ ^^J
1116       (sort-rule \string"#1\string" \string"#2\string")}%
1117   }
1118 \else
1119   \newcommand*\GlsAddSortRule[2]{%
1120     \glsnxindywarning\GlsAddSortRule}
1121 \fi
```

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

```
1122 \ifglxindy
1123   \def\@xdyrequiredstyles{tex}
1124 \fi
```

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

```
1125 \ifglxindy
1126   \newcommand*\GlsAddXdyStyle[1]{%
1127     \edef\@xdyrequiredstyles{\@xdyrequiredstyles,#1}}%
1128 \else
1129   \newcommand*\GlsAddXdyStyle[1]{%
1130     \glsnnoxindywarning\GlsAddXdyStyle}
1131 \fi
```

`\GlsSetXdyStyles` Reset the list of required styles

```
1132 \ifglxindy
1133   \newcommand*\GlsSetXdyStyles[1]{%
1134     \edef\@xdyrequiredstyles{#1}}
1135 \else
1136   \newcommand*\GlsSetXdyStyles[1]{%
1137     \glsnnoxindywarning\GlsSetXdyStyles}
1138 \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` no longer available. Now provide a command that does nothing (in case it's been patched).

```
1139 \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.

```
1140 \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.

```
1141 \ifglxindy
1142   \newcommand*\GlsSetXdyLanguage[2][\glsdefaulttype]{%
1143     \ifglossaryexists{#1}{%
1144       \expandafter\def\csname @xdy@#1@language\endcsname{#2}%
1145     }{%
1146       \PackageError{glossaries}{Can't set language type for
1147         glossary type '#1' --- no such glossary}{%
1148         You have specified a glossary type that doesn't exist}}
1149 \else
1150   \newcommand*\GlsSetXdyLanguage[2][]{%
1151     \glsnnoxindywarning\GlsSetXdyLanguage}
1152 \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.

```
1153 \def\@gls@codepage#1#2{}
```

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

```
1154 \ifglxindy
1155   \newcommand*{\GlsSetXdyCodePage}[1]{%
1156     \renewcommand*{\gls@codepage}{#1}%
1157   }
```

Suggested by egreg:

```
1158   \AtBeginDocument{%
1159     \ifx\gls@codepage\@empty
1160       \@ifpackageloaded{fontspec}{\def\gls@codepage{utf8}}{}%
1161     \fi
1162   }
1163 \else
1164   \newcommand*{\GlsSetXdyCodePage}[1]{%
1165     \glsnoxindywarning\GlsSetXdyCodePage}
1166 \fi
```

`\@xdylettergroups` Store letter group definitions.

```
1167 \ifglxindy
1168   \ifglx@xindy@glsnumbers
1169     \def\@xdylettergroups{(define-letter-group
1170       \string"glslnumbers\string"^^J\space\space\space
1171       :prefixes (\string"0\string" \string"1\string"
1172       \string"2\string" \string"3\string" \string"4\string"
1173       \string"5\string" \string"6\string" \string"7\string"
1174       \string"8\string" \string"9\string")^^J\space\space\space
1175       :before \string"@glsfirstletter\string")}
1176   \else
1177     \def\@xdylettergroups{}
1178   \fi
1179 \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.

```
1180   \newcommand*\GlsAddLetterGroup[2]{%
1181     \expandafter\toks@\expandafter{\@xdylettergroups}%
1182     \protected@edef\@xdylettergroups{\the\toks@^^J%
1183     (define-letter-group \string"#1\string"^^J\space\space\space#2)}%
1184   }
```

1.5 Loops and conditionals

`\forallglossaries` 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.

```
1185 \newcommand*\forallglossaries[3][\@glo@types]{%
1186   \@for#2:=#1\do{\ifx#2\@empty\else#3\fi}%
1187 }
```

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

`\forall glsentries [⟨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.

```
1188 \newcommand*\forallglsentries[3][\glsdefaulttype]{%
1189   \edef\@glo@list{\csname glolist@#1\endcsname}%
1190   \@for#2:=\@glo@list\do
1191     {%
1192       \ifdefempty{#2}{-}{#3}%
1193     }%
1194 }
```

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

`\forall allglsentries [⟨glossary list⟩] {⟨cmd⟩} {⟨code⟩}`

Within `\forall allglsentries`, the current glossary type is given by `\@this@glo@`.

```
1195 \newcommand*\forallallglsentries[3][\@glo@types]{%
1196   \expandafter\forallglossaries\expandafter[1]{\@this@glo@}%
1197   {%
1198     \forallglsentries[\@this@glo@]{#2}{#3}%
1199   }%
1200 }
```

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

`\if glossaryexists {⟨type⟩} {⟨true-text⟩} {⟨false-text⟩}`

where `⟨type⟩` is the glossary's label.

```
1201 \newcommand{\ifglossaryexists}[3]{%
1202   \ifcsundef{@glo@type@#1@out}{#3}{#2}%
1203 }
```

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

`\if glsentryexists {⟨label⟩} {⟨true text⟩} {⟨false text⟩}`

where $\langle label \rangle$ is the entry's label.

```
1204 \newcommand{\ifglstryexists}[3]{%
1205   \ifcsundef{glo@#1@name}{#3}{#2}%
1206 }
```

$\backslash\text{ifglused}$ To determine if given glossary entry has been used in the document text yet use:

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

where $\langle label \rangle$ is the entry's label. If true it will do $\langle true\ text \rangle$ otherwise it will do $\langle false\ text \rangle$.

```
1207 \newcommand*{\ifglused}[3]{\ifbool{glo@#1@flag}{#2}{#3}}
```

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

$\backslash\text{glstryexists}$ $\backslash\text{glstryexists}\{<label>\}\{<code>\}$

Generate an error if entry specified by $\langle label \rangle$ doesn't exist, otherwise do $\langle code \rangle$.

```
1208 \newcommand{\glstryexists}[2]{%
1209   \ifglstryexists{#1}{#2}{%
1210     \PackageError{glossaries}{Glossary entry ‘#1’ has not been
1211     defined}{You need to define a glossary entry before you
1212     can use it.}}%
1213 }
```

$\backslash\text{glstrynoexists}$ $\backslash\text{glstrynoexists}\{<label>\}\{<code>\}$

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

```
1214 \newcommand{\glstrynoexists}[2]{%
1215   \ifglstryexists{#1}{%
1216     \PackageError{glossaries}{Glossary entry ‘#1’ has already
1217     been defined}{}}{#2}%
1218 }
```

$\backslash\text{ifglshaschildren}$ $\backslash\text{ifglshaschildren}\{<label>\}\{<true\ part>\}\{<false\ part>\}$

```
1219 \newcommand{\ifglshaschildren}[3]{%
1220   \glstryexists{#1}%
1221   {%
1222     \def\do@glshaschildren{#3}%
1223     \expandafter\forglstry\expandafter[\csname glo@#1@type\endcsname]
1224     {\glo@label}%
1225     {%
1226       \letcs\glo@parent{glo@\glo@label @parent}%
1227       \ifthenelse{\equal{#1}{\glo@parent}}{%
1228         {%
1229           \def\do@glshaschildren{#2}%
1230           \@endfortrue
```

```

1231      }%
1232      {}%
1233      }%
1234      \do@glshaschildren
1235      }%
1236  }

\ifglshasparent \ifglshaschildren{<label>}{<true part>}{<false part>}
1237 \newcommand{\ifglshasparent}[3]{%
1238   \glsdoifexists{#1}%
1239   {%
1240     \ifcseempty{glo@#1@parent}{#3}{#2}%
1241   }%
1242 }

\ifglshasdesc \ifglshasdesc{<label>}{<true part>}{<false part>}
1243 \newcommand*\ifglshasdesc[3]{%
1244   \ifcseempty{glo@#1@desc}%
1245   {#3}%
1246   {#2}%
1247 }

\ifglsdessuppressed \ifglsdessuppressed{<label>}{<true part>}{<false part>} Does <true part>
if the description is just \nopostdesc otherwise does <false part>.
1248 \newcommand*\ifglsdessuppressed[3]{%
1249   \ifcsequal{glos@#1@desc}{@no@post@desc}%
1250   {#2}%
1251   {#3}%
1252 }

\ifglshassymbol \ifglshassymbol{<label>}{<true part>}{<false part>}
1253 \newcommand*\ifglshassymbol[3]{%
1254   \ifcseempty{glo@#1@symbol}%
1255   {#3}%
1256   {%
1257     \expandafter\ifx\csname glo@#1@symbol\endcsname\@gls@default@value
1258     #3%
1259   \else
1260     #2%
1261   \fi
1262   }%
1263 }

\ifglshaslong \ifglshaslong{<label>}{<true part>}{<false part>}
1264 \newcommand*\ifglshaslong[3]{%
1265   \ifcseempty{glo@#1@long}%
1266   {#3}%
1267   {%

```

```

1268 \expandafter\ifx\csname glo@#1@long\endcsname\@gls@default@value
1269 #3%
1270 \else
1271 #2%
1272 \fi
1273 }%
1274 }

```

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

```

1275 \newcommand*{\ifglshasshort}[3]{%
1276 \ifcsempy{glo@#1@short}%
1277 {#3}%
1278 {%
1279 \expandafter\ifx\csname glo@#1@short\endcsname\@gls@default@value
1280 #3%
1281 \else
1282 #2%
1283 \fi
1284 }%
1285 }

```

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`

```

1286 \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.

```

1287 \newcommand*{\@gls@provide@newglossary{%
1288 \protected@write\@auxout{}\string\providecommand\string\@newglossary[4]{}%
    Only need to do this once.
1289 \let\@gls@provide@newglossary\relax
1290 }

```

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

```

1291 \newcommand*{\defglentryfmt}[2][\glsdefaulttype]{%
1292 \csgdef{gls@#1@entryfmt}{#2}%
1293 }

```

`\gls@doentryfmt`

```

1294 \newcommand*{\gls@doentryfmt}[1]{\csuse{gls@#1@entryfmt}}

```

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`

```
1295 \newcommand*{\newglossary}[5][glg]{%
1296   \ifglossaryexists{#2}%
1297   {%
1298     \PackageError{glossaries}{Glossary type ‘#2’ already exists}{%
1299       You can’t define a new glossary called ‘#2’ because it already
1300       exists}%
1301   }%
1302   {%
      Check if default has been set
1303     \ifundef\glsdefaulttype
1304     {%
1305       \gdef\glsdefaulttype{#2}%
1306     }{%
      Add this to the list of glossary types:
1307     \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.
1308     \expandafter\gdef\csname glolist@#2\endcsname{,}%
      Store details of this new glossary type:
1309     \expandafter\def\csname @glotype@#2in\endcsname{#3}%
1310     \expandafter\def\csname @glotype@#2out\endcsname{#4}%
1311     \expandafter\def\csname @glotype@#2title\endcsname{#5}%
1312     \@gls@provide@newglossary
1313     \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.

```

1314 \ifcsundef{gls@#2@entryfmt}%
1315 {%
1316     \defglssentryfmt[#2]{\glssentryfmt}%
1317 }%
1318 {}%

```

Define sort counter if required:

```

1319 \@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.)

```

1320 \@ifnextchar[{\@gls@setcounter{#2}}%
1321     {\@gls@setcounter{#2}[\glscounter]}}%
1322 }

```

`\altnewglossary`

```

1323 \newcommand*{\altnewglossary}[3]{%
1324     \newglossary[#2-glg]{#1}{#2-gls}{#2-glo}{#3}%
1325 }

```

Only define new glossaries in the preamble:

```

1326 \@onlypreamble{\newglossary}

```

Only define new glossaries before `\makeglossaries`

```

1327 \@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 \TeX , `\@newglossary` simply ignores its arguments.

`\@newglossary`

```

1328 \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`

```

1329 \def\@gls@setcounter#1[#2]{%
1330     \expandafter\def\csname @glotype@#1@counter\endcsname{#2}%

```

Add counter to xindy list, if not already added:

```

1331     \ifglsxindy
1332         \GlsAddXdyCounters{#2}%
1333     \fi
1334 }

```

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

`\@gls@getcounter`

```

1335 \newcommand*{\@gls@getcounter}[1]{%
1336     \csname @glotype@#1@counter\endcsname
1337 }

```

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

```
1338 \glsdefmain
```

Define the “acronym” glossaries if required.

```
1339 \@gls@do@acronymsdef
```

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

```
1340 \@gls@do@symbolsdef
```

```
1341 \@gls@do@numbersdef
```

```
1342 \@gls@do@indexdef
```

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.

```
1343 \define@key{glossentry}{name}{%
```

```
1344 \def\@glo@name{#1}}%
```

```
1345 }
```

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`.

```
1346 \define@key{glossentry}{description}{%
```

```
1347 \def\@glo@desc{#1}}%
```

```
1348 }
```

descriptionplural

```
1349 \define@key{glossentry}{descriptionplural}{%
```

```
1350 \def\@glo@descplural{#1}}%
```

```
1351 }
```

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 `\langle name \rangle \langle description \rangle`.

```
1352 \define@key{glossentry}{sort}{%
```

```
1353 \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.

```
1354 \define@key{glossentry}{text}{%
1355 \def\@glo@text{#1}%
1356 }
```

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.

```
1357 \define@key{glossentry}{plural}{%
1358 \def\@glo@plural{#1}%
1359 }
```

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.

```
1360 \define@key{glossentry}{first}{%
1361 \def\@glo@first{#1}%
1362 }
```

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.

```
1363 \define@key{glossentry}{firstplural}{%
1364 \def\@glo@firstplural{#1}%
1365 }
```

`\@gls@default@value`

```
1366 \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).

```
1367 \define@key{glossentry}{symbol}{%
1368 \def\@glo@symbol{#1}%
1369 }
```

symbolplural

```
1370 \define@key{glossentry}{symbolplural}{%
1371 \def\@glo@symbolplural{#1}%
1372 }
```

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

```
1373 \define@key{glossentry}{type}{%
1374 \def\@glo@type{#1}}
```

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

```
1375 \define@key{glossentry}{counter}{%
1376 \ifcsundef{c@#1}%
1377 {%
1378 \PackageError{glossaries}%
1379 {There is no counter called ‘#1’}%
1380 {%
1381 The counter key should have the name of a valid counter
1382 as its value%
1383 }%
1384 }%
1385 {%
1386 \def\@glo@counter{#1}%
1387 }%
1388 }
```

see The see key specifies a list of cross-references

```
1389 \define@key{glossentry}{see}{%
1390 \gls@checkseeallowed
1391 \def\@glo@see{#1}%
1392 \@glo@seeautonumberlist
1393 }
```

gls@checkseeallowed

```
1394 \newcommand*{\gls@checkseeallowed}{%
1395 \PackageError{glossaries}%
1396 {‘see’ key may only be used after \string\makeglossaries}%
1397 {You must use \string\makeglossaries\space before defining
1398 any entries that have a ‘see’ key}%
1399 }
```

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

```
1400 \define@key{glossentry}{parent}{%
1401 \def\@glo@parent{#1}}
```

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

```
1402 \define@choicekey{glossentry}{nonumberlist}[\val\nr]{true,false}[true]{%
1403 \ifcase\nr\relax
1404 \def\@glo@prefix{\glsnonextpages}%
1405 \else
1406 \def\@glo@prefix{\glsnextpages}%
1407 }
```

```

1407 \fi
1408 }

```

Define some generic user keys. (6 ought to be enough!)

user1

```

1409 \define@key{glossentry}{user1}{%
1410 \def\@glo@useri{#1}%
1411 }

```

user2

```

1412 \define@key{glossentry}{user2}{%
1413 \def\@glo@userii{#1}%
1414 }

```

user3

```

1415 \define@key{glossentry}{user3}{%
1416 \def\@glo@useriii{#1}%
1417 }

```

user4

```

1418 \define@key{glossentry}{user4}{%
1419 \def\@glo@useriv{#1}%
1420 }

```

user5

```

1421 \define@key{glossentry}{user5}{%
1422 \def\@glo@userv{#1}%
1423 }

```

user6

```

1424 \define@key{glossentry}{user6}{%
1425 \def\@glo@uservi{#1}%
1426 }

```

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.

```

1427 \define@key{glossentry}{short}{%
1428 \def\@glo@short{#1}%
1429 }

```

shortplural This key is provided for use by \newacronym.

```

1430 \define@key{glossentry}{shortplural}{%
1431 \def\@glo@shortpl{#1}%
1432 }

```

long This key is provided for use by \newacronym.

```

1433 \define@key{glossentry}{long}{%
1434 \def\@glo@long{#1}%
1435 }

```

longplural This key is provided for use by \newacronym.

```
1436 \define@key{glossentry}{longplural}{%
1437   \def\@glo@longpl{#1}%
1438 }
```

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

```
1439 \newcommand*{\@glsnname}{%
1440   \PackageError{glossaries}{name key required in
1441     \string\newglossaryentry\space for entry '\@glo@label'}{You
1442     haven't specified the entry name}}
```

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

```
1443 \newcommand*{\@glsnodelsc}{%
1444   \PackageError{glossaries}
1445   {%
1446     description key required in \string\newglossaryentry\space
1447     for entry '\@glo@label'%
1448   }%
1449   {%
1450     You haven't specified the entry description%
1451   }%
1452 }%
```

\@glsdefaultplural Now obsolete. Don't use.

```
1453 \newcommand*{\@glsdefaultplural}{{}}
```

s@missingnumberlist Define a command to generate warning when numberlist not set.

```
1454 \newcommand*{\@gls@missingnumberlist}[1]{%
1455   ??%
1456   \ifglssavenumberlist
1457     \GlossariesWarning{Missing number list for entry '#1'.
1458       Maybe makeglossaries + rerun required.}%
1459   \else
1460     \PackageError{glossaries}%
1461     {Package option 'savenumberlist=true' required.}%
1462     {%
1463       You must use the 'savenumberlist' package option
1464       to reference location lists.%
1465     }%
1466   \fi
1467 }
```

\@glsdefaultsort Define command to set default sort.

```
1468 \newcommand*{\@glsdefaultsort}{\@glo@name}
```

\gls@level Register to increment entry levels.

```
1469 \newcount\gls@level
```

@gls@noexpand@field

```
1470 \newcommand{\@gls@noexpand@field}[3]{%
1471   \expandafter\global\expandafter
1472     \let\csname glo@#1@#2\endcsname#3%
1473 }
```

gls@noexpand@fields

```
1474 \newcommand{\@gls@noexpand@fields}[4]{%
1475   \ifcsdef{gls@assign@#3@field}
1476     {%
1477       \ifdefequal{#4}{\@gls@default@value}%
1478       {%
1479         \edef\@gls@value{\expandonce{#1}}%
1480         \csuse{gls@assign@#3@field}{#2}{\@gls@value}%
1481       }%
1482     }%
1483     \csuse{gls@assign@#3@field}{#2}{#4}%
1484   }%
1485 }%
1486 {%
1487   \ifdefequal{#4}{\@gls@default@value}%
1488   {%
1489     \edef\@gls@value{\expandonce{#1}}%
1490     \@gls@noexpand@field{#2}{#3}{\@gls@value}%
1491   }%
1492   {%
1493     \@gls@noexpand@field{#2}{#3}{#4}%
1494   }%
1495 }%
1496 }
```

\@gls@expand@field

```
1497 \newcommand{\@gls@expand@field}[3]{%
1498   \expandafter
1499     \protected@xdef\csname glo@#1@#2\endcsname{#3}%
1500 }
```

@gls@expand@fields

```
1501 \newcommand{\@gls@expand@fields}[4]{%
1502   \ifcsdef{gls@assign@#3@field}
1503     {%
1504       \ifdefequal{#4}{\@gls@default@value}%
1505       {%
1506         \edef\@gls@value{\expandonce{#1}}%
1507         \csuse{gls@assign@#3@field}{#2}{\@gls@value}%
1508       }%
1509     }%
1510     \expandafter\@gls@startswitexpandonce#4\relax\relax\gls@endcheck
```

```

1511      {%
1512      \@@gls@expand@field{#2}{#3}{#4}%
1513      }%
1514      {%
1515      \csuse{gls@assign@#3@field}{#2}{#4}%
1516      }%
1517      }%
1518      }%
1519      {%
1520      \ifdefequal{#4}{\@gls@default@value}%
1521      {%
1522      \@@gls@expand@field{#2}{#3}{#1}%
1523      }%
1524      {%
1525      \@@gls@expand@field{#2}{#3}{#4}%
1526      }%
1527      }%
1528      }

```

startswithexpandonce

```

1529 \def\@gls@expandonce{\expandonce}
1530 \def\@gls@startswithexpandonce#1#2\gls@endcheck#3#4{%
1531   \def\@gls@tmp{#1}%
1532   \ifdefequal{\@gls@expandonce}{\@gls@tmp}{#3}{#4}%
1533 }

```

`\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.

```
1534 \let\gls@assign@field\@gls@expand@fields
```

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

```

1535 \newcommand*\glsexpandfields{%
1536   \let\gls@assign@field\@gls@expand@fields
1537 }

```

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

```

1538 \newcommand*\glsnoexpandfields{%
1539   \let\gls@assign@field\@gls@noexpand@fields
1540 }

```

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

```

1541 \newrobustcmd{\newglossaryentry}[2]{%
    Check to see if this glossary entry has already been defined:
1542   \glsdoifnoexists{#1}%
1543   {%
1544     \gls@defglossaryentry{#1}{#2}%
1545   }%
1546 }

```

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

```

1547 \newrobustcmd{\provideglossaryentry}[2]{%
1548   \ifglsentryexists{#1}%
1549   {%
1550   }%
1551   \gls@defglossaryentry{#1}{#2}%
1552 }%
1553 }
1554 \@onlypreamble{\provideglossaryentry}

```

`\new@glossaryentry` For use in document environment.

```

1555 \newrobustcmd{\new@glossaryentry}[2]{%
1556   \ifundef\@gls@deffile
1557   {%
1558     \global\newwrite\@gls@deffile
1559     \immediate\openout\@gls@deffile=\jobname.glsdefs
1560   }%
1561   {%
1562     \ifglsentryexists{#1}{}%
1563     {%
1564       \gls@defglossaryentry{#1}{#2}%
1565     }%
1566     \@gls@writedef{#1}%
1567   }
1568 \AtBeginDocument
1569 {
1570   \makeatletter
1571   \InputIfFileExists{\jobname.glsdefs}{\{\}}%
1572   \makeatother
1573   \let\newglossaryentry\new@glossaryentry
1574 }
1575 \AtEndDocument{\ifdef\@gls@deffile{\closeout\@gls@deffile}\{\}}
1576 %   \end{macrocode}
1577 %\end{macro}
1578 %
1579 %\begin{macro}{\@gls@writedef}
1580 %\changes{3.10a}{2013-10-13}{new}
1581 % Writes glossary entry definition to \cs{@gls@deffile}.
1582 %   \begin{macrocode}

```

```

1583 \newcommand*{\@gls@writedef}[1]{%
1584   \immediate\write\@gls@deffile
1585   {%
1586     \string\ifglsentryexists{#1}\}\expandafter\@gobble\string\%^~J%
1587     \expandafter\@gobble\string\{\}\expandafter\@gobble\string\%^~J%
1588     \string\gls@defglossaryentry{#1}\expandafter\@gobble\string\%^~J%
1589     \expandafter\@gobble\string\{\}\expandafter\@gobble\string\%%
1590   }%

  Write key value information:

1591   \@for\@gls@map:=\@gls@keymap\do
1592   {%
1593     \edef\glo@value{\expandafter\expandonce
1594       \csname glo@#1\expandafter\@secondoftwo\@gls@map\endcsname}%
1595     \@onelevel@sanitize\glo@value
1596     \immediate\write\@gls@deffile
1597     {%
1598       \expandafter\@firstoftwo\@gls@map
1599       =\expandafter\@gobble\string\{\glo@value\expandafter\@gobble\string\},%
1600       \expandafter\@gobble\string\%%
1601     }%
1602   }%

  Provide hook:

1603   \gls@writedefhook
1604   \immediate\write\@gls@deffile
1605   {%
1606     \expandafter\@gobble\string\%^~J%
1607     \expandafter\@gobble\string\}\expandafter\@gobble\string\%^~J%
1608     \expandafter\@gobble\string\}\expandafter\@gobble\string\%%
1609   }%
1610 }

```

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

```

1611 \newcommand*{\@gls@keymap}{%
1612   {name}{name},%
1613   {sort}{sortvalue},% unescaped sort value
1614   {type}{type},%
1615   {first}{first},%
1616   {firstplural}{firstpl},%
1617   {text}{text},%
1618   {plural}{plural},%
1619   {description}{desc},%
1620   {descriptionplural}{descplural},%
1621   {symbol}{symbol},%
1622   {symbolplural}{symbolplural},%
1623   {user1}{useri},%
1624   {user2}{userii},%
1625   {user3}{useriii},%

```

```

1626 {user4}{useriv},%
1627 {user5}{userv},%
1628 {user6}{uservi},%
1629 {long}{long},%
1630 {longplural}{longpl},%
1631 {short}{short},%
1632 {shortplural}{shortpl},%
1633 {counter}{counter},%
1634 {parent}{parent}%
1635 }

```

`\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.

```

1636 \newcommand*{\glsaddkey}{\@ifstar\sglsaddkey\glsaddkey}

```

Starred version switches on expansion for this key.

```

1637 \newcommand*{\@sglsaddkey}[1]{%
1638   \key@ifundefined{glossentry}{#1}%
1639   {%
1640     \expandafter\newcommand\expandafter*\expandafter
1641       {\csname gls@assign@#1@field\endcsname}[2]{%
1642         \@gls@expand@field{##1}{#1}{##2}%
1643       }%
1644   }%
1645   }%
1646   \@glsaddkey{#1}%
1647 }

```

Unstarred version doesn't override default expansion.

```

1648 \newcommand*{\@glsaddkey}[7]{%

```

Check the specified key doesn't already exist.

```

1649   \key@ifundefined{glossentry}{#1}%
1650   {%

```

Set up the key.

```

1651     \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
1652     \appto\@gls@keymap{,{#1}{#1}}%

```

Set the default value.

```

1653     \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%

```

Assignment code.

```

1654     \appto\@newglossaryentryposthook{%
1655       \letcs{\@glo@tmp}{@glo@#1}%
1656       \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
1657     }%

```

Define the no-link commands.

```

1658 \newcommand*{#3}[1]{\csuse{glo@##1@#1}}%
1659 \newcommand*{#4}[1]{%
1660 \letcs{\@glo@text}{glo@##1@#1}%
1661 \xmakefirstuc\@glo@text
1662 }%

```

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

```

1663 \ifcsdef{@gls@user@#1@}%
1664 {%
1665 \PackageError{glossaries}%
1666 {Can't define '\string#5' as helper command
1667 '\expandafter\string\csname @gls@user@#1@\endcsname' already exists}%
1668 }%
1669 }%
1670 {%
1671 \newrobustcmd*{#5}{\@ifstar{\csuse{@sgls@user@#1}}{\csuse{@gls@user@#1}}}%
1672 \expandafter\newcommand\expandafter*\expandafter
1673 {\csname @sgls@user@#1\endcsname}[1][{%
1674 \csuse{@gls@user@#1}[hyper=false,##1]%
1675 }%
1676 \expandafter\newcommand\expandafter*\expandafter
1677 {\csname @gls@user@#1\endcsname}[2][{%
1678 \new@ifnextchar[%
1679 {\csuse{@gls@user@#1@}{##1}{##2}}%
1680 {\csuse{@gls@user@#1@}{##1}{##2}[]}}%
1681 \csdef{@gls@user@#1@}##1##2[##3]{%
1682 \glsdoifexists{##2}%
1683 {%
1684 \edef\@glo@type{\glsentrytype{##2}}%
1685 \@gls@link[##1]{##2}{#3{##2}##3}%
1686 }%
1687 }%
1688 }%

```

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

```

1689 \ifcsdef{@Gls@user@#1@}%
1690 {%
1691 \PackageError{glossaries}%
1692 {Can't define '\string#6' as helper command
1693 '\expandafter\string\csname @Gls@user@#1@\endcsname' already exists}%
1694 }%
1695 }%
1696 {%
1697 \newrobustcmd*{#6}{\@ifstar{\csuse{@sGls@user@#1}}{\csuse{@Gls@user@#1}}}%
1698 \expandafter\newcommand\expandafter*\expandafter
1699 {\csname @sGls@user@#1\endcsname}[1][{%
1700 \csuse{@Gls@user@#1}[hyper=false,##1]%
1701 }%
1702 \expandafter\newcommand\expandafter*\expandafter

```

```

1703     {\csname @GLs@user@#1\endcsname}[2] [] {%
1704         \new@ifnextchar [%
1705             {\csuse{@GLs@user@#1@}{##1}{##2}}%
1706             {\csuse{@GLs@user@#1@}{##1}{##2} []}}%
1707     \csdef{@GLs@user@#1@}##1##2[##3] {%
1708         \glsdoifexists{##2}%
1709         {%
1710             \edef\@glo@type{\glsentrytype{##2}}%
1711             \@gls@link[##1]{##2}{#4{##2}##3}%
1712         }%
1713     }%
1714 }%

```

Finally the all caps version:

```

1715     \ifcsdef{@GLS@user@#1@}%
1716     {%
1717         \PackageError{glossaries}%
1718         {Can't define '\string#7' as helper command
1719         '\expandafter\string\csname @GLS@user@#1@\endcsname' already exists}%
1720         {}%
1721     }%
1722     {%
1723         \newrobustcmd*{#7}{\@ifstar{\csuse{@sGLS@user@#1}}{\csuse{@GLS@user@#1}}}%
1724         \expandafter\newcommand\expandafter*\expandafter
1725         {\csname @sGLS@user@#1\endcsname}[1] [] {%
1726             \csuse{@GLS@user@#1}[hyper=false,##1]%
1727         }%
1728         \expandafter\newcommand\expandafter*\expandafter
1729         {\csname @GLS@user@#1\endcsname}[2] [] {%
1730             \new@ifnextchar [%
1731                 {\csuse{@GLS@user@#1@}{##1}{##2}}%
1732                 {\csuse{@GLS@user@#1@}{##1}{##2} []}}%
1733         \csdef{@GLS@user@#1@}##1##2[##3] {%
1734             \glsdoifexists{##2}%
1735             {%
1736                 \edef\@glo@type{\glsentrytype{##2}}%
1737                 \@gls@link[##1]{##2}{\mfirstucMakeUppercase{#3{##2}##3}}%
1738             }%
1739         }%
1740     }%
1741 }%
1742 {%
1743     \PackageError{glossaries}{Key '#1' already exists}{}%
1744 }%
1745 }

```

\glswritedefhook

```

1746 \newcommand*\glswritedefhook{}

```

`\gls@assign@desc`

```
1747 \newcommand*{\gls@assign@desc}[1]{%
1748   \gls@assign@field{#1}{desc}{\@glo@desc}%
1749   \gls@assign@field{\@glo@desc}{#1}{descplural}{\@glo@descplural}%
1750 }
```

`\longnewglossaryentry`

```
1751 \newcommand{\longnewglossaryentry}[3]{%
1752   \glsdoifnoexists{#1}%
1753   {%
1754     \bgroup
1755     \let\@org@newglossaryentryprehook\@newglossaryentryprehook
1756     \long\def\@newglossaryentryprehook{%
1757       \long\def\@glo@desc{#3\leavevmode\unskip\nopostdesc}%
1758       \@org@newglossaryentryprehook
1759     }%
1760     \renewcommand*{\gls@assign@desc}[1]{%
1761       \global\cslet{glo@#1@desc}{\@glo@desc}%
1762       \global\cslet{glo@#1@descplural}{\@glo@desc}%
1763     }
1764     \gls@defglossaryentry{#1}{#2}%
1765   \egroup
1766 }
1767 }
```

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

```
1768 \@onlypreamble{\longnewglossaryentry}
```

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

```
1769 \newcommand{\longprovideglossaryentry}[3]{%
1770   \ifglentryexists{#1}{}%
1771   {\longnewglossaryentry{#1}{#2}{#3}}%
1772 }
1773 \@onlypreamble{\longprovideglossaryentry}
```

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

Defines a new entry without checking if it already exists.

```
1774 \newcommand{\gls@defglossaryentry}[2]{%
```

Store label

```
1775   \def\@glo@label{#1}%
```

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

```
1776   \let\glslabel\@glo@label
```

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

```

1777 \let\@glo@name\@glsnname
1778 \let\@glo@desc\@glsnodesc

1779 \let\@glo@descplural\@gls@default@value
1780 \let\@glo@type\@gls@default@value
1781 \let\@glo@symbol\@gls@default@value

1782 \let\@glo@symbolplural\@gls@default@value
1783 \let\@glo@text\@gls@default@value
1784 \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.)

```

1785 \let\@glo@first\@gls@default@value
1786 \let\@glo@firstplural\@gls@default@value

```

Set the default sort:

```

1787 \let\@glo@sort\@gls@default@value

```

Set the default counter:

```

1788 \let\@glo@counter\@gls@default@value

1789 \def\@glo@see{}%

1790 \def\@glo@parent{}%

1791 \def\@glo@prefix{}%

1792 \def\@glo@useri{}%
1793 \def\@glo@userii{}%
1794 \def\@glo@useriii{}%
1795 \def\@glo@useriv{}%
1796 \def\@glo@userv{}%
1797 \def\@glo@uservi{}%

1798 \def\@glo@short{}%
1799 \def\@glo@shortpl{}%
1800 \def\@glo@long{}%
1801 \def\@glo@longpl{}%

```

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

```

1802 \@newglossaryentryprehook

```

Extract key-val information from third parameter:

```

1803 \setkeys{glossentry}{#2}%

```

Check there is a default glossary.

```
1804 \ifundef\glsdefaulttype
1805 {%
1806   \PackageError{glossaries}%
1807     {No default glossary type (have you used ‘nomain’?)}%
1808     {If you use package option ‘nomain’ you must define
1809      a new glossary before you can define entries}%
1810 }%
1811 {}%
```

Assign type. This must be fully expandable

```
1812 \gls@assign@field{\glsdefaulttype}{#1}{type}{\@glo@type}%
1813 \edef\@glo@type{\glsentrytype{#1}}%
```

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

```
1814 \ifcsundef{glolist@\@glo@type}%
1815 {%
1816   \PackageError{glossaries}%
1817     {Glossary type ‘\@glo@type’ has not been defined}%
1818     {You need to define a new glossary type, before making entries
1819      in it}%
1820 }%
1821 {%
1822   \protected@edef\glolist@{\csname glolist@\@glo@type\endcsname}%
1823   \expandafter\xdef\csname glolist@\@glo@type\endcsname{\glolist@{#1},}%
1824 }%
```

Initialise level to 0.

```
1825 \gls@level=0\relax
```

Has this entry been assigned a parent?

```
1826 \ifx\@glo@parent\@empty
```

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

```
1827 \expandafter\gdef\csname glo@#1@parent\endcsname{}%
1828 \else
```

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

```
1829 \ifthenelse{\equal{#1}{\@glo@parent}}%
1830 {%
1831   \PackageError{glossaries}{Entry ‘#1’ can’t be its own parent}{}%
1832   \def\@glo@parent{}%
1833   \expandafter\gdef\csname glo@#1@parent\endcsname{}%
1834 }%
1835 {}%
```

Check the parent exists:

```
1836 \ifglentryexists{\@glo@parent}%
1837 {%
```

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

```
1838 \expandafter\xdef\csname glo@#1@parent\endcsname{\@glo@parent}%
```

Determine level.

```
1839      \gls@level=\csname glo@\@glo@parent @level\endcsname\relax
1840      \advance\gls@level by 1\relax
```

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

```
1841      \ifx\@glo@name\@gls@name
1842      \expandafter\let\expandafter\@glo@name
1843      \csname glo@\@glo@parent @name\endcsname
```

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

```
1844      \ifx\@glo@plural\@gls@default@value
1845      \expandafter\let\expandafter\@glo@plural
1846      \csname glo@\@glo@parent @plural\endcsname
1847      \fi
1848      \fi
1849      }%
1850      {%
```

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

```
1851      \PackageError{glossaries}%
1852      {%
1853      Invalid parent '\@glo@parent'
1854      for entry '#1' - parent doesn't exist%
1855      }%
1856      {%
1857      Parent entries must be defined before their children%
1858      }%
1859      \def\@glo@parent{%
1860      \expandafter\gdef\csname glo@#1@parent\endcsname{%
1861      }%
1862      }%
1863      \fi
```

Set the level for this entry

```
1864      \expandafter\xdef\csname glo@#1@level\endcsname{\number\gls@level}%
```

Define commands associated with this entry:

```
1865      \gls@assign@field{\@glo@name}{#1}{sortvalue}{\@glo@sort}%
1866      \letcs\@glo@sort{glo@#1@sortvalue}%
1867      \gls@assign@field{\@glo@name}{#1}{text}{\@glo@text}%
1868      \expandafter\gls@assign@field\expandafter
1869      {\csname glo@#1@text\endcsname\glspluralsuffix}%
1870      {#1}{plural}{\@glo@plural}%
1871      \expandafter\gls@assign@field\expandafter
1872      {\csname glo@#1@text\endcsname}%
1873      {#1}{first}{\@glo@first}%
```

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

```
1874      \ifx\@glo@first\@gls@default@value
```

```

1875 \expandafter\gls@assign@field\expandafter
1876 {\csname glo@#1@plural\endcsname}%
1877 {#1}{firstpl}{\@glo@firstplural}%
1878 \else
1879 \expandafter\gls@assign@field\expandafter
1880 {\csname glo@#1@first\endcsname\glspluralsuffix}%
1881 {#1}{firstpl}{\@glo@firstplural}%
1882 \fi

1883 \ifcsundef{@glo@type@\@glo@type @counter}%
1884 {%
1885 \def\@glo@defaultcounter{\glscounter}%
1886 }%
1887 {%
1888 \letcs\@glo@defaultcounter{@glo@type@\@glo@type @counter}%
1889 }%
1890 \gls@assign@field{\@glo@defaultcounter}{#1}{counter}{\@glo@counter}%
1891 \gls@assign@field{}{#1}{useri}{\@glo@useri}%
1892 \gls@assign@field{}{#1}{userii}{\@glo@userii}%
1893 \gls@assign@field{}{#1}{useriii}{\@glo@useriii}%
1894 \gls@assign@field{}{#1}{useriv}{\@glo@useriv}%
1895 \gls@assign@field{}{#1}{userv}{\@glo@userv}%
1896 \gls@assign@field{}{#1}{uservi}{\@glo@uservi}%
1897 \gls@assign@field{}{#1}{short}{\@glo@short}%
1898 \gls@assign@field{}{#1}{shortpl}{\@glo@shortpl}%
1899 \gls@assign@field{}{#1}{long}{\@glo@long}%
1900 \gls@assign@field{}{#1}{longpl}{\@glo@longpl}%
1901 \ifx\@glo@name\@glsnoname
1902 \@glsnoname
1903 \let\@glo@name\@gls@default@value
1904 \fi
1905 \gls@assign@field{}{#1}{name}{\@glo@name}%

```

Set default numberlist if not defined:

```

1906 \ifcsundef{glo@#1@numberlist}%
1907 {%
1908 \csxdef{glo@#1@numberlist}{\noexpand\@gls@missingnumberlist{\@glo@label}}%
1909 }%
1910 {}%

```

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.

```

1911 \def\@glo@@desc{\@glo@first}%
1912 \ifx\@glo@desc\@glo@@desc
1913 \let\@glo@desc\@glo@first
1914 \fi
1915 \ifx\@glo@desc\@glsnodesc
1916 \@glsnodesc
1917 \let\@glo@desc\@gls@default@value

```

```

1918 \fi
1919 \gls@assign@desc{#1}%

```

Set the sort key for this entry:

```

1920 \@gls@defsort{\@glo@type}{#1}%

1921 \def\@glo@@symbol{\@glo@text}%
1922 \ifx\@glo@symbol\@glo@@symbol
1923 \let\@glo@symbol\@glo@text
1924 \fi
1925 \gls@assign@field{\relax}{#1}{symbol}{\@glo@symbol}%
1926 \expandafter
1927 \gls@assign@field\expandafter
1928 {\csname glo@#1@symbol\endcsname}
1929 {#1}{symbolplural}{\@glo@symbolplural}%

```

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

```

1930 \expandafter\gdef\csname glo@#1@flagfalse\endcsname{%
1931 \expandafter\global\expandafter
1932 \let\csname ifglo@#1@flag\endcsname\iffalse
1933 }%
1934 \expandafter\gdef\csname glo@#1@flagtrue\endcsname{%
1935 \expandafter\global\expandafter
1936 \let\csname ifglo@#1@flag\endcsname\iftrue
1937 }%
1938 \csname glo@#1@flagfalse\endcsname

```

Sort out any cross-referencing if required.

```

1939 \ifx\@glo@see\@empty
1940 \else
1941 \protected@edef\@do@glsee{%
1942 \noexpand\@gls@fixbraces\noexpand\@glo@list\@glo@see
1943 \noexpand\@nil
1944 \noexpand\expandafter\noexpand\@glsee\noexpand\@glo@list{#1}}%
1945 \@do@glsee
1946 \fi

```

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

```

1947 \do@glo@storeentry{#1}%

```

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

```

1948 \@newglossaryentryposthook
1949 }

```

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

```

1950 \newcommand*{\@newglossaryentryprehook}{}

```

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

```

1951 \newcommand*{\@newglossaryentryposthook}{}

```

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

```

1952 \newcommand*{\glsmoveentry}[2]{%
1953   \edef\glo@type{\csname glo@#1@type\endcsname}%
1954   \def\glo@list{,}%
1955   \forlslentries[\glo@type]{\glo@label}%
1956   {%
1957     \ifthenelse{\equal{\glo@label}{#1}}{\eappto\glo@list{\glo@label,}}%
1958   }%
1959   \cslet{glolist@\glo@type}{\glo@list}%
1960   \csdef{glo@#1@type}{#2}%
1961 }
```

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

```

1962 \ifglxindy
1963   \newcommand*{@glossaryentryfield}{\string\glossentry}
1964 \else
1965   \newcommand*{@glossaryentryfield}{\string\glossentry}
1966 \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.)

```

1967 \ifglxindy
1968   \newcommand*{@glossarysubentryfield}{%
1969     \string\subglossentry}
1970 \else
1971   \newcommand*{@glossarysubentryfield}{%
1972     \string\subglossentry}
1973 \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. The result is stored in `\glo@<label>@entry`, where `<label>` is the entry's label. (This doesn't include any formatting or location information.)

```

1974 \newcommand{\@glo@storeentry}[1]{%
  Escape special characters in the label:
1975   \def\@glo@label{#1}%
1976   \@gls@checkmkidxchars\@glo@label
  Get the sort string and escape any special characters
1977   \protected@edef\@glo@sort{\csname glo@#1@sort\endcsname}%
1978   \@gls@checkmkidxchars\@glo@sort
```

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

```

1979 \@gls@checkmkidxchars\@glo@prefix

Get the parent, if one exists
1980 \edef\@glo@parent{\csname glo@#1@parent\endcsname}%

Write the information to the glossary file.
1981 \ifglxsindy

Store using xindy syntax.
1982 \ifx\@glo@parent\@empty

Entry doesn't have a parent
1983 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
1984 (\string"\@glo@sort\string" %
1985 \string"\@glo@prefix\@glossaryentryfield{\@glo@label}\string") %
1986 }%
1987 \else

Entry has a parent
1988 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
1989 \csname glo@\@glo@parent @index\endcsname
1990 (\string"\@glo@sort\string" %
1991 \string"\@glo@prefix\@glossarysubentryfield
1992 {\csname glo@#1@level\endcsname}{\@glo@label}\string") %
1993 }%
1994 \fi
1995 \else

Store using makeindex syntax.
1996 \ifx\@glo@parent\@empty

Sanitize \@glo@prefix
1997 \@onelevel@sanitize\@glo@prefix

Entry doesn't have a parent
1998 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
1999 \@glo@sort\@gls@actualchar\@glo@prefix
2000 \@glossaryentryfield{\@glo@label}%
2001 }%
2002 \else

Entry has a parent
2003 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
2004 \csname glo@\@glo@parent @index\endcsname\@gls@levelchar
2005 \@glo@sort\@gls@actualchar\@glo@prefix
2006 \@glossarysubentryfield
2007 {\csname glo@#1@level\endcsname}{\@glo@label}%
2008 }%
2009 \fi
2010 \fi
2011 }

```

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`

```
2012 \AtBeginDocument{%
2013   \ifpackageloaded{amsmath}%
2014   {\let\gls@ifnotmeasuring\@gls@ifnotmeasuring}%
2015   }{%
2016 }
2017 \newcommand*{\@gls@ifnotmeasuring}[1]{%
2018   \ifmeasuring@
2019   \else
2020     #1%
2021   \fi
2022 }
2023 \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.

```
2024 \newcommand*{\glsreset}[1]{%
2025   \gls@ifnotmeasuring
2026   {%
2027     \glsdoifexists{#1}%
2028     {%
2029       \expandafter\global\csname glo@#1@flagfalse\endcsname
2030     }%
2031   }%
2032 }
```

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

```
2033 \newcommand*{\glslocalreset}[1]{%
2034   \gls@ifnotmeasuring
2035   {%
2036     \glsdoifexists{#1}%
2037     {%
2038       \expandafter\let\csname ifglo@#1@flag\endcsname\iffalse
2039     }%
2040   }%
2041 }
```

`\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.

```
2042 \newcommand*{\glsunset}[1]{%
```

```

2043 \gls@ifnotmeasuring
2044 {%
2045     \glsdoifexists{#1}%
2046     {%
2047         \expandafter\global\csname glo@#1@flagtrue\endcsname
2048     }%
2049 }%
2050 }

```

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

```

2051 \newcommand*\glslocalunset}[1]{%
2052     \gls@ifnotmeasuring
2053     {%
2054         \glsdoifexists{#1}%
2055         {%
2056             \expandafter\let\csname ifglo@#1@flag\endcsname\iftrue
2057         }%
2058     }%
2059 }

```

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

Syntax: `\glsresetall[⟨glossary-list⟩]`

`\glsresetall`

```

2060 \newcommand*\glsresetall}[1][\@glo@types]{%
2061     \forallglsentries[#1]{\@glsentry}%
2062     {%
2063         \glsreset{\@glsentry}%
2064     }%
2065 }

```

As above, but with only a local effect:

`\glslocalresetall`

```

2066 \newcommand*\glslocalresetall}[1][\@glo@types]{%
2067     \forallglsentries[#1]{\@glsentry}%
2068     {%
2069         \glslocalreset{\@glsentry}%
2070     }%
2071 }

```

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

Syntax: `\glsunsetall[⟨glossary-list⟩]`

`\glsunsetall`

```

2072 \newcommand*\glsunsetall}[1][\@glo@types]{%
2073     \forallglsentries[#1]{\@glsentry}%
2074     {%
2075         \glsunset{\@glsentry}%
2076     }%
2077 }

```

As above, but with only a local effect:

```
\glslocalunsetall
```

```
2078 \newcommand*{\glslocalunsetall}[1][\@gls@types]{%
2079   \forallglsentries[#1]{\@glsentry}%
2080   {%
2081     \glslocalunset{\@glsentry}%
2082   }%
2083 }
```

1.9 Loading files containing glossary entries

Glossary entries can be defined in an external file. These external files can contain `\newglossaryentry` and `\newacronym` commands.¹

```
\loadglsentries[⟨type⟩]{⟨filename⟩}
```

This command will input the file using `\input`. The optional argument specifies to which glossary the entries should be assigned if they haven't used the type key. If the optional argument is not specified, the default glossary is used. Only those entries used in the document (via `\glslink`, `\gls`, `\glspl` and uppercase variants or `\glsadd` and `\glsaddall` will appear in the glossary). The mandatory argument is the filename (with or without `.tex` extension).

```
\loadglsentries
```

```
2084 \newcommand*{\loadglsentries}[2][\@gls@default]{%
2085   \let\@gls@default\glsdefaulttype
2086   \def\glsdefaulttype{#1}\input{#2}%
2087   \let\glsdefaulttype\@gls@default
2088 }
```

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

```
2089 \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 `\glsformat`. By default this just displays the link text “as is”.

```
\glsformat
```

```
2090 \newcommand*{\glsformat}[1]{#1}
```

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

`\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

```
2091 \newcommand*{\glsentryfmt}{%
2092   \@gls@default@entryfmt\glsdisplayfirst\glsdisplay
2093 }
```

Format that provides backwards compatibility:

```
2094 \newcommand*{\@gls@default@entryfmt}[2]{%
2095   \ifdefempty\glscustomtext
2096     {%
2097       \glsifplural
2098       {%
```

Plural form

```
2099       \glscapscase
2100       {%
```

Don't adjust case

```
2101       \ifglsused\glslabel
2102       {%
```

Subsequent use

```
2103         #2{\glsentryplural{\glslabel}}%
2104         {\glsentrydescplural{\glslabel}}%
2105         {\glsentrysymbolplural{\glslabel}}{\glsinsert}%
2106     }%
2107     {%
```

First use

```
2108         #1{\glsentryfirstplural{\glslabel}}%
2109         {\glsentrydescplural{\glslabel}}%
2110         {\glsentrysymbolplural{\glslabel}}{\glsinsert}%
2111     }%
2112 }%
2113 {%
```

Make first letter upper case

```
2114     \ifglsused\glslabel
2115     {%
```

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

```
2116     \ifbool{glscompatible-3.07}%
2117     {%
2118       \protected@edef\@glo@etext{%
2119         #2{\glsentryplural{\glslabel}}%
2120         {\glsentrydescplural{\glslabel}}%
```

```

2121         {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2122     \xmakefirstuc\@glo@etext
2123 }%
2124 {%
2125     #2{\Glsentryplural{\glslabel}}%
2126     {\glsentrydescplural{\glslabel}}%
2127     {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2128 }%
2129 }%
2130 {%

```

First use

```

2131     \ifbool{glscompatible-3.07}%
2132     {%
2133         \protected@edef\@glo@etext{%
2134             #1{\glsentryfirstplural{\glslabel}}%
2135             {\glsentrydescplural{\glslabel}}%
2136             {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2137         \xmakefirstuc\@glo@etext
2138     }%
2139     {%
2140         #1{\Glsentryfirstplural{\glslabel}}%
2141         {\glsentrydescplural{\glslabel}}%
2142         {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2143     }%
2144 }%
2145 }%
2146 {%

```

Make all upper case

```

2147     \ifglsused\glslabel
2148     {%

```

Subsequent use

```

2149         \mfirstucMakeUppercase{#2{\glsentryplural{\glslabel}}%
2150         {\glsentrydescplural{\glslabel}}%
2151         {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2152     }%
2153     {%

```

First use

```

2154         \mfirstucMakeUppercase{#1{\glsentryfirstplural{\glslabel}}%
2155         {\glsentrydescplural{\glslabel}}%
2156         {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2157     }%
2158 }%
2159 }%
2160 {%

```

Singular form

```

2161     \glscapscase
2162     {%

```

Don't adjust case

```
2163      \ifglsused\glslabel
2164      {%
```

Subsequent use

```
2165      #2{\glsentrytext{\glslabel}}%
2166      {\glsentrydesc{\glslabel}}%
2167      {\glsentrysymbol{\glslabel}}{\glsinsert}%
2168      }%
2169      {%
```

First use

```
2170      #1{\glsentryfirst{\glslabel}}%
2171      {\glsentrydesc{\glslabel}}%
2172      {\glsentrysymbol{\glslabel}}{\glsinsert}%
2173      }%
2174      }%
2175      {%
```

Make first letter upper case

```
2176      \ifglsused\glslabel
2177      {%
```

Subsequent use

```
2178      \ifbool{glscompatible-3.07}%
2179      {%
2180      \protected@edef\@glo@etext{%
2181      #2{\glsentrytext{\glslabel}}%
2182      {\glsentrydesc{\glslabel}}%
2183      {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2184      \xmakefirstuc\@glo@etext
2185      }%
2186      {%
2187      #2{\Glsentrytext{\glslabel}}%
2188      {\glsentrydesc{\glslabel}}%
2189      {\glsentrysymbol{\glslabel}}{\glsinsert}%
2190      }%
2191      }%
2192      {%
```

First use

```
2193      \ifbool{glscompatible-3.07}%
2194      {%
2195      \protected@edef\@glo@etext{%
2196      #1{\glsentryfirst{\glslabel}}%
2197      {\glsentrydesc{\glslabel}}%
2198      {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2199      \xmakefirstuc\@glo@etext
2200      }%
2201      {%
2202      #1{\Glsentryfirst{\glslabel}}%
```

```

2203             {\glsentrydesc{\glslabel}}%
2204             {\glsentrysymbol{\glslabel}}{\glsinsert}%
2205         }%
2206     }%
2207 }%
2208 {%

    Make all upper case
2209     \ifglsused\glslabel
2210     {%

        Subsequent use
2211         \mfirstucMakeUppercase{#2{\glsentrytext{\glslabel}}%
2212             {\glsentrydesc{\glslabel}}%
2213             {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2214         }%
2215         {%

            First use
2216             \mfirstucMakeUppercase{#1{\glsentryfirst{\glslabel}}%
2217                 {\glsentrydesc{\glslabel}}%
2218                 {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2219             }%
2220         }%
2221     }%
2222 }%
2223 {%

    Custom text provided in \glsdisp
2224     \ifglsused{\glslabel}%
2225     {%

        Subsequent use
2226         #2{\glscustomtext}%
2227         {\glsentrydesc{\glslabel}}%
2228         {\glsentrysymbol{\glslabel}}{%
2229     }%
2230     {%

        First use
2231         #1{\glscustomtext}%
2232         {\glsentrydesc{\glslabel}}%
2233         {\glsentrysymbol{\glslabel}}{%
2234     }%
2235     }%
2236 }

```

`\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.

```

2237 \newcommand*{\glsgenentryfmt}{%
2238     \ifdefempty\glscustomtext

```

```

2239  {%
2240    \glsifplural
2241  {%

  Plural form
2242    \glscapscase
2243  {%

  Don't adjust case
2244    \ifglsused\glslabel
2245  {%

  Subsequent use
2246    \glsentryplural{\glslabel}\glsinsert
2247  }%
2248  {%

  First use
2249    \glsentryfirstplural{\glslabel}\glsinsert
2250  }%
2251  }%
2252  {%

  Make first letter upper case
2253    \ifglsused\glslabel
2254  {%

  Subsequent use.
2255    \Glsentryplural{\glslabel}\glsinsert
2256  }%
2257  {%

  First use
2258    \Glsentryfirstplural{\glslabel}\glsinsert
2259  }%
2260  }%
2261  {%

  Make all upper case
2262    \ifglsused\glslabel
2263  {%

  Subsequent use
2264    \mfirstucMakeUppercase
2265    {\glsentryplural{\glslabel}\glsinsert}%
2266  }%
2267  {%

  First use
2268    \mfirstucMakeUppercase
2269    {\glsentryfirstplural{\glslabel}\glsinsert}%
2270  }%
2271  }%

```

2272 }%
2273 {%

Singular form

2274 \glscapscase
2275 {%

Don't adjust case

2276 \ifglused\glslabel
2277 {%

Subsequent use

2278 \glentrytext{\glslabel}\glinsert
2279 }%
2280 {%

First use

2281 \glentryfirst{\glslabel}\glinsert
2282 }%
2283 }%
2284 {%

Make first letter upper case

2285 \ifglused\glslabel
2286 {%

Subsequent use

2287 \Glentrytext{\glslabel}\glinsert
2288 }%
2289 {%

First use

2290 \Glentryfirst{\glslabel}\glinsert
2291 }%
2292 }%
2293 {%

Make all upper case

2294 \ifglused\glslabel
2295 {%

Subsequent use

2296 \mfirstucMakeUppercase{\glentrytext{\glslabel}\glinsert}%
2297 }%
2298 {%

First use

2299 \mfirstucMakeUppercase{\glentryfirst{\glslabel}\glinsert}%
2300 }%
2301 }%
2302 }%
2303 }%
2304 {%

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

```
2305     \glscustomtext\glsinsert
2306   }%
2307 }
```

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

```
2308 \newcommand*{\glsngenacfmt}{%
2309   \ifdefempty\glscustomtext
2310     {%
2311       \ifglsused\glslabel
2312         {%
```

Subsequent use:

```
2313       \glsifplural
2314         {%
```

Subsequent plural form:

```
2315       \glscapscase
2316         {%
```

Subsequent plural form, don't adjust case:

```
2317       \acronymfont{\glsentryshortpl{\glslabel}}\glsinsert
2318     }%
2319   {%
```

Subsequent plural form, make first letter upper case:

```
2320       \acronymfont{\Glsentryshortpl{\glslabel}}\glsinsert
2321     }%
2322   {%
```

Subsequent plural form, all caps:

```
2323       \mfirstucMakeUppercase
2324       {\acronymfont{\glsentryshortpl{\glslabel}}\glsinsert}%
2325     }%
2326   }%
2327 }
```

Subsequent singular form

```
2328       \glscapscase
2329         {%
```

Subsequent singular form, don't adjust case:

```
2330       \acronymfont{\glsentryshort{\glslabel}}\glsinsert
2331     }%
2332   {%
```

Subsequent singular form, make first letter upper case:

```
2333       \acronymfont{\Glsentryshort{\glslabel}}\glsinsert
2334     }%
2335   {%
```

Subsequent singular form, all caps:

```
2336      \mfirstucMakeUppercase
2337      {\acronymfont{\glsentryshort{\glslabel}}\glsinsert}%
2338      }%
2339      }%
2340      }%
2341      {%
```

First use:

```
2342      \glsifplural
2343      {%
```

First use plural form:

```
2344      \glscapscase
2345      {%
```

First use plural form, don't adjust case:

```
2346      \genplacrfullformat{\glslabel}{\glsinsert}%
2347      }%
2348      {%
```

First use plural form, make first letter upper case:

```
2349      \Genplacrfullformat{\glslabel}{\glsinsert}%
2350      }%
2351      {%
```

First use plural form, all caps:

```
2352      \mfirstucMakeUppercase
2353      {\genplacrfullformat{\glslabel}{\glsinsert}}%
2354      }%
2355      }%
2356      {%
```

First use singular form

```
2357      \glscapscase
2358      {%
```

First use singular form, don't adjust case:

```
2359      \genacrfullformat{\glslabel}{\glsinsert}%
2360      }%
2361      {%
```

First use singular form, make first letter upper case:

```
2362      \Genacrfullformat{\glslabel}{\glsinsert}%
2363      }%
2364      {%
```

First use singular form, all caps:

```
2365      \mfirstucMakeUppercase
2366      {\genacrfullformat{\glslabel}{\glsinsert}}%
2367      }%
2368      }%
2369      }%
```

```

2370 }%
2371 {%
    User supplied text.
2372 \glscustomtext
2373 }%
2374 }

```

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

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

```

2375 \newcommand*{\genacrfullformat}[2]{%
2376   \glentrylong{#1}#2\space
2377   (\protect\firstacronymfont{\glentryshort{#1}})%
2378 }

```

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

As above but makes the first letter upper case.

```

2379 \newcommand*{\Genacrfullformat}[2]{%
2380   \protected@edef\gls@text{\genacrfullformat{#1}{#2}}%
2381   \xmakefirstuc\gls@text
2382 }

```

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

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

```

2383 \newcommand*{\genplacrfullformat}[2]{%
2384   \glentrylongpl{#1}#2\space
2385   (\protect\firstacronymfont{\glentryshortpl{#1}})%
2386 }

```

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

As above but makes the first letter upper case.

```

2387 \newcommand*{\Genplacrfullformat}[2]{%
2388   \protected@edef\gls@text{\genplacrfullformat{#1}{#2}}%
2389   \xmakefirstuc\gls@text
2390 }

```

`\glsdisplayfirst` Deprecated. Kept for backward compatibility.

```

2391 \newcommand*{\glsdisplayfirst}[4]{#1#4}

```

`\glsdisplay` Deprecated. Kept for backward compatibility.

```
2392 \newcommand*{\glsdisplay}[4]{#1#4}
```

`\defglsdisplay` Deprecated. Kept for backward compatibility.

```
2393 \newcommand*{\defglsdisplay}[2][\glsdefaulttype]{%
2394   \GlossariesWarning{\string\defglsdisplay\space is now obsolete.^^J
2395   Use \string\defglsentryfmt\space instead}%
2396   \expandafter\def\csname gls@#1@display\endcsname##1##2##3##4{#2}%
2397   \edef\@gls@doentrydef{%
2398     \noexpand\defglsentryfmt[#1]{%
2399       \noexpand\ifcsdef{gls@#1@displayfirst}%
2400       {%
2401         \noexpand\@gls@default@entryfmt
2402         {\noexpand\csuse{gls@#1@displayfirst}}
2403         {\noexpand\csuse{gls@#1@display}}}%
2404       }%
2405       {%
2406         \noexpand\@gls@default@entryfmt
2407         {\noexpand\glsdisplayfirst}
2408         {\noexpand\csuse{gls@#1@display}}}%
2409       }%
2410     }%
2411   }%
2412   \@gls@doentrydef
2413 }
```

`\defglsdisplayfirst` Deprecated. Kept for backward compatibility.

```
2414 \newcommand*{\defglsdisplayfirst}[2][\glsdefaulttype]{%
2415   \GlossariesWarning{\string\defglsdisplayfirst\space is now obsolete.^^J
2416   Use \string\defglsentryfmt\space instead}%
2417   \expandafter\def\csname gls@#1@displayfirst\endcsname##1##2##3##4{#2}%
2418   \edef\@gls@doentrydef{%
2419     \noexpand\defglsentryfmt[#1]{%
2420       \noexpand\ifcsdef{gls@#1@display}%
2421       {%
2422         \noexpand\@gls@default@entryfmt
2423         {\noexpand\csuse{gls@#1@displayfirst}}
2424         {\noexpand\csuse{gls@#1@display}}}%
2425       }%
2426       {%
2427         \noexpand\@gls@default@entryfmt
2428         {\noexpand\csuse{gls@#1@displayfirst}}%
2429         {\noexpand\glsdisplay}
2430       }%
2431     }%
2432   }%
2433   \@gls@doentrydef
2434 }
```

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.

```
2435 \define@key{glslink}{counter}{%
2436   \ifcsundef{c@#1}%
2437   {%
2438     \PackageError{glossaries}%
2439     {There is no counter called ‘#1’}%
2440     {%
2441       The counter key should have the name of a valid counter
2442       as its value%
2443     }%
2444   }%
2445   {%
2446     \def\@gls@counter{#1}%
2447   }%
2448 }
```

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.

```
2449 \define@key{glslink}{format}{%
2450 \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.

```
2451 \define@boolkey{glslink}{hyper}[true]{}
```

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.

```
2452 \define@boolkey{glslink}{local}[true]{}
```

Syntax:

`\glslink[options]{label}{text}`

Display $\langle text \rangle$ in the document, and add the entry information for $\langle label \rangle$ 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*[\langle options \rangle]{\langle label \rangle}{\langle text \rangle}`

which is equivalent to `\glslink[hyper=false,\langle options \rangle]{\langle label \rangle}{\langle text \rangle}`

First determine whether or not we are using the starred version:

`\glslink`

```
2453 \newrobustcmd*{\glslink}{%
2454   \ifstar\@sgls@link\@gls@link
2455 }
```

`\@sgls@link` The starred version of `\glslink` calls the unstarred version with hyperlinks disabled.

```
2456 \newcommand*{\@sgls@link}[1] [] {\@gls@link[hyper=false,#1]}
```

`\@gls@link` The unstarred version of `\glslink` checks for the existence of the term. 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.

```
2457 \newcommand*{\@gls@link}[3] [] {%
2458   \ifglsentryexists{#2}%
2459   {%
2460     \@gls@link[#1]{#2}{#3}%
2461   }{%
2462     \PackageError{glossaries}{Glossary entry ‘#2’ has not been
2463     defined}{You need to define a glossary entry before you
2464     can use it.}%

```

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

```
2465   \glstextformat{#3}%
2466 }%
2467 }
```

`\@gls@link`

```
2468 \def\@gls@link[#1]#2#3{%
```

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

```
2469   \leavevmode
2470   \def\glslabel{#2}%
2471   \def\glsnumberformat{glsnumberformat}%
2472   \edef\@gls@counter{\csname glo@#2@counter\endcsname}%

```

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

```

2473 \edef\gls@type{\csname glo@#2@type\endcsname}%
2474 \expandafter\DTLifinlist\expandafter
2475   {\gls@type}{\@gls@nohyperlist}%
2476   {%
2477     \KV@glslink@hyperfalse
2478   }%
2479   {%
2480     \KV@glslink@hypertrue
2481   }%
2482   \setkeys{glslink}{#1}%

```

Store the entry's counter in \theglsentrycounter

```

2483   \@gls@saveentrycounter

```

Define sort key if necessary:

```

2484   \@gls@setsort{#2}%

```

```

2485   \@do@wrglossary{#2}%
2486   \ifKV@glslink@hyper
2487     \@glslink{\glolinkprefix#2}{\glstextformat{#3}}%
2488   \else

```

```

2489     \glstextformat{#3}%
2490   \fi
2491 }

```

\glolinkprefix

```

2492 \newcommand*{\glolinkprefix}{glo:}

```

\glsentrycounter Set default value of entry counter

```

2493 \def\glsentrycounter{\glscounter}%

```

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

```

2494 \newcommand*{\@gls@saveentrycounter}{%
2495   \def\@gls@Hcounter{%

```

Are we using equation counter?

```

2496   \ifthenelse{\equal{\@gls@counter}{equation}}{%
2497     {

```

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

```

2498   \ifcsundef{xatlevel@}%
2499   {%
2500     \edef\theglsentrycounter{\expandafter\noexpand
2501       \csname the\@gls@counter\endcsname}%
2502   }%
2503   {%
2504     \ifx\xatlevel@\@empty
2505       \edef\theglsentrycounter{\expandafter\noexpand
2506         \csname the\@gls@counter\endcsname}%

```

```

2507     \else
2508         \savecounters@
2509         \advance\c@equation by 1\relax
2510         \edef\theglentrycounter{\csname the\@gls@counter\endcsname}%

```

Check if hyperref version of this counter

```

2511     \ifcsundef{theH\@gls@counter}%
2512     {%
2513         \def\@gls@Hcounter{\theglentrycounter}%
2514     }%
2515     {%
2516         \def\@gls@Hcounter{\csname theH\@gls@counter\endcsname}%
2517     }%
2518     \protected@edef\theHglentrycounter{\@gls@Hcounter}%
2519     \restorecounters@
2520 \fi
2521 }%
2522 }%
2523 {%

```

Not using equation counter so no special measures:

```

2524     \edef\theglentrycounter{\expandafter\noexpand
2525         \csname the\@gls@counter\endcsname}%
2526 }%

```

Check if hyperref version of this counter

```

2527 \ifx\@gls@Hcounter\@empty
2528     \ifcsundef{theH\@gls@counter}%
2529     {%
2530         \def\theHglentrycounter{\theglentrycounter}%
2531     }%
2532     {%
2533         \protected@edef\theHglentrycounter{\expandafter\noexpand
2534             \csname theH\@gls@counter\endcsname}%
2535     }%
2536 \fi
2537 }

```

`\@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.

```

2538 \def\@set@glo@numformat#1#2#3#4{%
2539     \expandafter\@glo@check@mkidxrangear#3\@nil
2540     \protected@edef#1{%
2541         \@glo@prefix setentrycounter[#4]{#2}%
2542         \expandafter\string\csname\@glo@suffix\endcsname
2543     }%

```

```

2544 \@gls@checkmkidxchars#1%
2545 }

```

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.

```

2546 \def\@glo@check@mkidxrangechar#1#2\@nil{%
2547 \if#1(\relax
2548   \def\@glo@prefix{(%}
2549   \if\relax#2\relax
2550     \def\@glo@suffix{glsnumberformat}%
2551   \else
2552     \def\@glo@suffix{#2}%
2553   \fi
2554 \else
2555   \if#1)\relax
2556     \def\@glo@prefix{)%}
2557   \if\relax#2\relax
2558     \def\@glo@suffix{glsnumberformat}%
2559   \else
2560     \def\@glo@suffix{#2}%
2561   \fi
2562 \else
2563   \def\@glo@prefix{}\def\@glo@suffix{#1#2}%
2564 \fi
2565 \fi}

```

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

```

2566 \newcommand*{\@gls@escbsdq}[1]{%
2567   \def\@gls@checkedmkidx{}%
2568   \let\gls@xdystring=#1\relax
2569   \@onelevel@sanitize\gls@xdystring
2570   \edef\do@gls@xdycheckbackslash{%
2571     \noexpand\@gls@xdycheckbackslash\gls@xdystring\noexpand\@nil
2572     \@backslashchar\@backslashchar\noexpand\null}%
2573   \do@gls@xdycheckbackslash
2574   \expandafter\@gls@updatechecked\@gls@checkedmkidx{\gls@xdystring}%
2575   \def\@gls@checkedmkidx{}%
2576   \expandafter\@gls@xdycheckquote\gls@xdystring\@nil""\null
2577   \expandafter\@gls@updatechecked\@gls@checkedmkidx{\gls@xdystring}%

```

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

```

2578   \@for\@gls@tmp:=\gls@protected@pagefmts\do
2579   {%
2580     \edef\@gls@sanitized@tmp{\expandafter\@gobble\string\\expandonce\@gls@tmp}%
2581     \@onelevel@sanitize\@gls@sanitized@tmp

```

```

2582 \edef\gls@dostsubst{%
2583 \noexpand\DTLsubstituteall\noexpand\gls@xdyststring
2584 {\@gls@sanitized@tmp}{\expandonce\@gls@tmp}%
2585 }%
2586 \gls@dostsubst
2587 }%

```

Assign to required control sequence

```

2588 \let#1=\gls@xdyststring
2589 }

```

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

`\gls@checkmkidxchars`

```

2590 \newcommand{\@gls@checkmkidxchars}[1]{%
2591 \ifglxsindy
2592 \@gls@escbsdq{#1}%
2593 \else
2594 \def\@gls@checkedmkidx{%
2595 \expandafter\@gls@checkquote#1\@nil""\null
2596 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2597 \def\@gls@checkedmkidx{%
2598 \expandafter\@gls@checkescquote#1\@nil""\null
2599 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2600 \def\@gls@checkedmkidx{%
2601 \expandafter\@gls@checkescactual#1\@nil"??\null
2602 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2603 \def\@gls@checkedmkidx{%
2604 \expandafter\@gls@checkactual#1\@nil??\null
2605 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2606 \def\@gls@checkedmkidx{%
2607 \expandafter\@gls@checkbar#1\@nil||\null
2608 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2609 \def\@gls@checkedmkidx{%
2610 \expandafter\@gls@checkescbar#1\@nil|||\null
2611 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2612 \def\@gls@checkedmkidx{%
2613 \expandafter\@gls@checklevel#1\@nil!!\null
2614 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2615 \fi
2616 }

```

Update the control sequence and strip trailing `\@nil`:

`\@gls@updatechecked`

```

2617 \def\@gls@updatechecked#1\@nil#2{\def#2{#1}}

```

`\@gls@tmpb` Define temporary token

```

2618 \newtoks\@gls@tmpb

```

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

```
2619 \def\@gls@checkquote#1"#2"#3\null{%
2620   \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2621   \toks@={#1}%
2622   \ifx\null#2\null
2623     \ifx\null#3\null
2624       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2625       \def\@gls@checkquote{\relax}%
2626     \else
2627       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2628         \@gls@quotechar\@gls@quotechar\@gls@quotechar}%
2629       \def\@gls@checkquote{\@gls@checkquote#3\null}%
2630     \fi
2631   \else
2632     \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2633       \@gls@quotechar\@gls@quotechar}%
2634     \ifx\null#3\null
2635       \def\@gls@checkquote{\@gls@checkquote#2""\null}%
2636     \else
2637       \def\@gls@checkquote{\@gls@checkquote#2"#3\null}%
2638     \fi
2639   \fi
2640   \@gls@checkquote
2641 }
```

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

```
2642 \def\@gls@checkescquote#1\"#2\"#3\null{%
2643   \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2644   \toks@={#1}%
2645   \ifx\null#2\null
2646     \ifx\null#3\null
2647       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2648       \def\@gls@checkescquote{\relax}%
2649     \else
2650       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2651         \@gls@quotechar\string\" \@gls@quotechar
2652         \@gls@quotechar\string\" \@gls@quotechar}%
2653       \def\@gls@checkescquote{\@gls@checkescquote#3\null}%
2654     \fi
2655   \else
2656     \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2657       \@gls@quotechar\string\" \@gls@quotechar}%
2658     \ifx\null#3\null
2659       \def\@gls@checkescquote{\@gls@checkescquote#2\"\" \null}%
2660     \else
2661       \def\@gls@checkescquote{\@gls@checkescquote#2\"#3\null}%
2662     \fi
2663   \fi
2664   \@gls@checkescquote
```

2665 }

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

```
2666 \def@gls@checkescactual#1\?#2\?#3\null{%
2667 \gls@tmpb=\expandafter{\gls@checkedmkidx}%
2668 \toks@={#1}%
2669 \ifx\null#2\null
2670 \ifx\null#3\null
2671 \edef@gls@checkedmkidx{\the@gls@tmpb\the\toks@}%
2672 \def@@gls@checkescactual{\relax}%
2673 \else
2674 \edef@gls@checkedmkidx{\the@gls@tmpb\the\toks@
2675 \gls@quotechar\string"\@gls@actualchar
2676 \gls@quotechar\string"\@gls@actualchar}%
2677 \def@@gls@checkescactual{\@gls@checkescactual#3\null}%
2678 \fi
2679 \else
2680 \edef@gls@checkedmkidx{\the@gls@tmpb\the\toks@
2681 \gls@quotechar\string"\@gls@actualchar}%
2682 \ifx\null#3\null
2683 \def@@gls@checkescactual{\@gls@checkescactual#2\?\?\null}%
2684 \else
2685 \def@@gls@checkescactual{\@gls@checkescactual#2\?#3\null}%
2686 \fi
2687 \fi
2688 @@gls@checkescactual
2689 }
```

\@gls@checkescbar Similarly for \||:

```
2690 \def@gls@checkescbar#1\||#2\||#3\null{%
2691 \gls@tmpb=\expandafter{\gls@checkedmkidx}%
2692 \toks@={#1}%
2693 \ifx\null#2\null
2694 \ifx\null#3\null
2695 \edef@gls@checkedmkidx{\the@gls@tmpb\the\toks@}%
2696 \def@@gls@checkescbar{\relax}%
2697 \else
2698 \edef@gls@checkedmkidx{\the@gls@tmpb\the\toks@
2699 \gls@quotechar\string"\@gls@encapchar
2700 \gls@quotechar\string"\@gls@encapchar}%
2701 \def@@gls@checkescbar{\@gls@checkescbar#3\null}%
2702 \fi
2703 \else
2704 \edef@gls@checkedmkidx{\the@gls@tmpb\the\toks@
2705 \gls@quotechar\string"\@gls@encapchar}%
2706 \ifx\null#3\null
2707 \def@@gls@checkescbar{\@gls@checkescbar#2\||\null}%
2708 \else
2709 \def@@gls@checkescbar{\@gls@checkescbar#2\||#3\null}%
2710 }
```

```

2710 \fi
2711 \fi
2712 \@@gls@checkescbar
2713 }

```

\@gls@checkesclevel Similarly for \!:

```

2714 \def\@gls@checkesclevel#1\!#2\!#3\null{%
2715 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2716 \toks@={#1}%
2717 \ifx\null#2\null
2718 \ifx\null#3\null
2719 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2720 \def\@@gls@checkesclevel{\relax}%
2721 \else
2722 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2723 \@gls@quotechar\string"\@gls@levelchar
2724 \@gls@quotechar\string"\@gls@levelchar}%
2725 \def\@@gls@checkesclevel{\@gls@checkesclevel#3\null}%
2726 \fi
2727 \else
2728 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2729 \@gls@quotechar\string"\@gls@levelchar}%
2730 \ifx\null#3\null
2731 \def\@@gls@checkesclevel{\@gls@checkesclevel#2\!\!\null}%
2732 \else
2733 \def\@@gls@checkesclevel{\@gls@checkesclevel#2\!#3\null}%
2734 \fi
2735 \fi
2736 \@@gls@checkesclevel
2737 }

```

\@gls@checkbar and for |:

```

2738 \def\@gls@checkbar#1|#2|#3\null{%
2739 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2740 \toks@={#1}%
2741 \ifx\null#2\null
2742 \ifx\null#3\null
2743 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2744 \def\@@gls@checkbar{\relax}%
2745 \else
2746 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2747 \@gls@quotechar\@gls@encapchar\@gls@quotechar\@gls@encapchar}%
2748 \def\@@gls@checkbar{\@gls@checkbar#3\null}%
2749 \fi
2750 \else
2751 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2752 \@gls@quotechar\@gls@encapchar}%
2753 \ifx\null#3\null
2754 \def\@@gls@checkbar{\@gls@checkbar#2||\null}%

```

```

2755 \else
2756 \def\@gls@checkbar{\@gls@checkbar#2|#3\null}%
2757 \fi
2758 \fi
2759 \@gls@checkbar
2760 }

```

\@gls@checklevel and for !:

```

2761 \def\@gls@checklevel#1!#2!#3\null{%
2762 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2763 \toks@={#1}%
2764 \ifx\null#2\null
2765 \ifx\null#3\null
2766 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2767 \def\@gls@checklevel{\relax}%
2768 \else
2769 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2770 \@gls@quotechar\@gls@levelchar\@gls@quotechar\@gls@levelchar}%
2771 \def\@gls@checklevel{\@gls@checklevel#3\null}%
2772 \fi
2773 \else
2774 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2775 \@gls@quotechar\@gls@levelchar}%
2776 \ifx\null#3\null
2777 \def\@gls@checklevel{\@gls@checklevel#2!!\null}%
2778 \else
2779 \def\@gls@checklevel{\@gls@checklevel#2!#3\null}%
2780 \fi
2781 \fi
2782 \@gls@checklevel
2783 }

```

\@gls@checkactual and for ?:

```

2784 \def\@gls@checkactual#1?#2?#3\null{%
2785 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2786 \toks@={#1}%
2787 \ifx\null#2\null
2788 \ifx\null#3\null
2789 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2790 \def\@gls@checkactual{\relax}%
2791 \else
2792 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2793 \@gls@quotechar\@gls@actualchar\@gls@quotechar\@gls@actualchar}%
2794 \def\@gls@checkactual{\@gls@checkactual#3\null}%
2795 \fi
2796 \else
2797 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2798 \@gls@quotechar\@gls@actualchar}%
2799 \ifx\null#3\null

```

```

2800     \def\@gls@checkactual{\@gls@checkactual#2??\null}%
2801     \else
2802     \def\@gls@checkactual{\@gls@checkactual#2?#3\null}%
2803     \fi
2804     \fi
2805     \@gls@checkactual
2806 }

```

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

```

2807 \def\@gls@xdycheckquote#1"#2"#3\null{%
2808   \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2809   \toks@={#1}%
2810   \ifx\null#2\null
2811     \ifx\null#3\null
2812       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2813       \def\@gls@xdycheckquote{\relax}%
2814     \else
2815       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2816         \string"\string"}%
2817       \def\@gls@xdycheckquote{\@gls@xdycheckquote#3\null}%
2818     \fi
2819   \else
2820     \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2821       \string"}%
2822     \ifx\null#3\null
2823       \def\@gls@xdycheckquote{\@gls@xdycheckquote#2""\null}%
2824     \else
2825       \def\@gls@xdycheckquote{\@gls@xdycheckquote#2"#3\null}%
2826     \fi
2827     \fi
2828     \@gls@xdycheckquote
2829 }

```

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

```

\@gls@xdycheckbackslash
2830 \edef\def\@gls@xdycheckbackslash{%
2831   \noexpand\def\noexpand\@gls@xdycheckbackslash##1\@backslashchar
2832   ##2\@backslashchar##3\noexpand\null{%
2833     \noexpand\@gls@tmpb=\noexpand\expandafter
2834     {\noexpand\@gls@checkedmkidx}%
2835     \noexpand\toks@={##1}%
2836     \noexpand\ifx\noexpand\null##2\noexpand\null
2837     \noexpand\ifx\noexpand\null##3\noexpand\null
2838     \noexpand\edef\noexpand\@gls@checkedmkidx{%
2839       \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@}%
2840     \noexpand\def\noexpand\@gls@xdycheckbackslash{\relax}%
2841     \noexpand\else
2842     \noexpand\edef\noexpand\@gls@checkedmkidx{%
2843       \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@

```

```

2844 \backslashchar\backslashchar\backslashchar\backslashchar}%
2845 \noexpand\def\noexpand\@gls@xdycheckbackslash{%
2846 \noexpand\@gls@xdycheckbackslash##3\noexpand\null}%
2847 \noexpand\fi
2848 \noexpand\else
2849 \noexpand\edef\noexpand\@gls@checkedmkidx{%
2850 \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
2851 \backslashchar\backslashchar}%
2852 \noexpand\ifx\noexpand\null##3\noexpand\null
2853 \noexpand\def\noexpand\@gls@xdycheckbackslash{%
2854 \noexpand\@gls@xdycheckbackslash##2\backslashchar
2855 \backslashchar\noexpand\null}%
2856 \noexpand\else
2857 \noexpand\def\noexpand\@gls@xdycheckbackslash{%
2858 \noexpand\@gls@xdycheckbackslash##2\backslashchar
2859 ##3\noexpand\null}%
2860 \noexpand\fi
2861 \noexpand\fi
2862 \noexpand\@gls@xdycheckbackslash
2863 }%
2864 }

```

Now go ahead and define \gls@xdycheckbackslash

```
2865 \def@gls@xdycheckbackslash
```

\glslink If \hyperlink is not defined \glslink ignores its first argument and just does the second argument, otherwise it is equivalent to \hyperlink.

```

2866 \ifcsundef{hyperlink}%
2867 {%
2868 \gdef@glslink#1#2{#2}%
2869 }%
2870 {%
2871 \gdef@glslink#1#2{\hyperlink{#1}{#2}}%
2872 }

```

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

```

2873 \newlength@gls@tmplen \ifcsundef{hypertarget}%
2874 {%
2875 \gdef@glstarget#1#2{#2}%
2876 }%
2877 {%
2878 \gdef@glstarget#1#2{%
2879 \settoheight@gls@tmplen{#2}%
2880 \raisebox@gls@tmplen{\hypertarget{#1}}{#2}%
2881 }%
2882 }

```

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

`\glsdisablehyper`

```
2883 \newcommand{\glsdisablehyper}{%
2884   \renewcommand*\@glslink[2]{##2}%
2885   \renewcommand*\@glstarget[2]{##2}%
2886 }
```

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

`\glsenablehyper`

```
2887 \newcommand{\glsenablehyper}{%
2888   \renewcommand*\@glslink[2]{\hyperlink{##1}{##2}}%
2889   \renewcommand*\@glstarget[2]{%
2890     \settoheight{\gls@tmplen}{##2}%
2891     \raisebox{\gls@tmplen}{\hypertarget{##1}{}}##2}}
```

Provide some convenience commands if not already defined:

```
2892 \providecommand{\@firstofthree}[3]{#1}
2893 \providecommand{\@secondofthree}[3]{#2}
2894 \providecommand{\@thirdofthree}[3]{#3}
```

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 if we are using the starred form:

`\gls`

```
2895 \newrobustcmd*{\gls}{\@ifstar\@sgls\@gls}
```

Define the starred form:

`\@sgls`

```
2896 \newcommand*{\@sgls}[1][\@gls[hyper=false,#1]]
```

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

\@gls

```
2897 \newcommand*{\@gls}[2] [] {%
2898   \new@ifnextchar[{\@gls@{#1}{#2}}{\@gls@{#1}{#2} []}%
2899 }
```

\@gls@ Read in the final optional argument:

```
2900 \def\@gls@#1#2[#3] {%
2901   \glsdoifexists{#2}%
2902   {%
2903     \edef\@glo@type{\glsentrytype{#2}}%
2904     Save options in \@gls@link@opts and label in \@gls@link@label
2905     \def\@gls@link@opts{#1}%
2906     \def\@gls@link@label{#2}%
2907     \def\glslabel{#2}%
2908     \let\glsifplural\@secondoftwo
2909     \let\glscapscase\@firstofthree
2910     \let\glscustomtext\@empty
2911     \def\glsinsert{#3}%
```

Determine what the link text should be (this is stored in \@glo@text)

```
2912     \def\@glo@text{\csname gls@\@glo@type @entryfmt\endcsname}%
2913     Call \@gls@link. If footnote package option has been used and the glossary
2914     type is \acronymtype, suppress hyperlink for first use. Likewise if the hyper-
2915     first=false package option is used.
2916     \ifglsused{#2}%
2917     {%
2918       \@gls@link[#1]{#2}{\@glo@text}%
2919       }%
2920       {%
2921         \gls@checkisacronymlist\@glo@type
2922         \ifthenelse
2923         {(\boolean{glsisacronymlist}\AND \boolean{glsacrfootnote})}
2924         \OR \NOT\boolean{glshyperfirst}
2925         {%
2926           \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
2927           }%
2928           {%
2929             \@gls@link[#1]{#2}{\@glo@text}%
2930             }%
2931             }%
2932             }%
```

Indicate that this entry has now been used

```
2929   \ifKV@glslink@local
2930     \glslocalunset{#2}%
2931   \else
2932     \glsunset{#2}%
```

```

2933 \fi
2934 }%
2935 }

```

`\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`

```

2936 \newrobustcmd*{\Gls}{\@ifstar\@sGls\@Gls}

```

Define the starred form:

```

2937 \newcommand*{\@sGls}[1][\@Gls[hyper=false,#1]]{

```

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

```

2938 \newcommand*{\@Gls}[2][\@Gls[hyper=false,#1]]{
2939 \new@ifnextchar{\@Gls@{#1}{#2}}{\@Gls@{#1}{#2}[]}%
2940 }

```

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

```

2941 \def\@Gls@#1#2[#3]{%
2942 \glsdoifexists{#2}%
2943 {%
2944 \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2945 \def\@gls@link@opts{#1}%
2946 \def\@gls@link@label{#2}%
2947 \def\@gls@label{#2}%

2948 \let\@glsifplural\@secondoftwo
2949 \let\@gls@scapscase\@secondofthree
2950 \let\@gls@customtext\@empty
2951 \def\@gls@insert{#3}%

```

Determine what the link text should be (this is stored in `\@glo@text`)

```

2952 \def\@glo@text{\csname gls@\@glo@type @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 `hyper-first=false` package option is used.

```

2953 \ifglsused{#2}%
2954 {%
2955 \@gls@link[#1]{#2}{\@glo@text}%
2956 }%
2957 {%
2958 \gls@checkisacronymlist\@glo@type
2959 \ifthenelse

```

```

2960      {%
2961      \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
2962      \OR \NOT\boolean{glshyperfirst}%
2963      }%
2964      {%
2965      \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
2966      }%
2967      {%
2968      \@gls@link[#1]{#2}{\@glo@text}%
2969      }%
2970      }%

```

Indicate that this entry has now been used

```

2971      \ifKV@glslink@local
2972      \glslocalunset{#2}%
2973      \else
2974      \glsunset{#2}%
2975      \fi
2976      }%
2977 }

```

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

\GLS

```

2978 \newrobustcmd*{\GLS}{\@ifstar\@sGLS\@GLS}

```

Define the starred form:

```

2979 \newcommand*{\@sGLS}[1][\@GLS[hyper=false,#1]]{

```

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

```

2980 \newcommand*{\@GLS}[2][\@GLS@#1]{
2981   \new@ifnextchar[\@GLS@{#1}{#2}]{\@GLS@{#1}{#2}[]}%
2982 }

```

\@GLS@ Read in the final optional argument:

```

2983 \def\@GLS@#1#2[#3]{%
2984   \glsdoifexists{#2}%
2985   {%
2986     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2987   \def\@gls@link@opts{#1}%
2988   \def\@gls@link@label{#2}%
2989   \def\glslabel{#2}%
2990   \let\glsifplural\@secondoftwo
2991   \let\glscapscase\@thirdofthree
2992   \let\glscustomtext\@empty
2993   \def\glsinsert{#3}%

```

Determine what the link text should be (this is stored in \@glo@text).

```

2994   \def\@glo@text{\csname gls@\@glo@type @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 hyper-
      first=false package option is used.
2995   \ifglsused{#2}%
2996   {%
2997     \@gls@link[#1]{#2}{\@glo@text}%
2998   }%
2999   {%
3000     \gls@checkisacronymlist\@glo@type
3001     \ifthenelse
3002     {%
3003       \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
3004       \OR \NOT\boolean{glshyperfirst}}{%
3005       \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
3006     }%
3007     {%
3008       \@gls@link[#1]{#2}{\@glo@text}%
3009     }%
3010   }%

```

Indicate that this entry has now been used

```

3011   \ifKV@glslink@local
3012   \glslocalunset{#2}%
3013   \else
3014   \glsunset{#2}%
3015   \fi
3016 }%
3017 }

```

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

\glspl

```

3018 \newrobustcmd*{\glspl}{\@ifstar\@sglspl\@glspl}

```

Define the starred form:

```

3019 \newcommand*{\@sglspl}[1][\@glspl[hyper=false,#1]]

```

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

```

3020 \newcommand*{\@glspl}[2][\@glspl@{#1}{#2}[]]{%
3021   \new@ifnextchar[\@glspl@{#1}{#2}]{\@glspl@{#1}{#2}[]}%
3022 }

```

\@glspl@ Read in the final optional argument:

```

3023 \def\@glspl@#1#2[#3]{%
3024   \glsdoifexists{#2}%
3025   {%
3026     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

3027 \def\@gls@link@opts{#1}%
3028 \def\@gls@link@label{#2}%

3029 \def\glslabel{#2}%
3030 \let\glsifplural\@firstoftwo
3031 \let\glscapscase\@firstofthree
3032 \let\glscustomtext\@empty
3033 \def\glsinsert{#3}%
3034 % Determine what the link text should be (this is stored in
3035 % \cs{@glo@text})
3036 %\changes{1.12}{2008 Mar 8}{now uses \cs{glsentrydescplural} and
3037 % \cs{glsentrysymbolplural} instead of \cs{glsentrydesc} and
3038 % \cs{glsentrysymbol}}
3039 %\changes{3.11a}{2013-10-15}{change to using \cs{glsentryfmt} style
3040 %commands}
3041 % \begin{macrocode}
3042 \def\@glo@text{\csname gls@\@glo@type @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.

```

3043 \ifglsused{#2}%
3044 {%
3045 \@gls@link[#1]{#2}{\@glo@text}%
3046 }%
3047 {%
3048 \gls@checkisacronymlist\@glo@type
3049 \ifthenelse
3050 {%
3051 \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
3052 \OR \NOT\boolean{glshyperfirst}}%
3053 {%
3054 \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
3055 }%
3056 {%
3057 \@gls@link[#1]{#2}{\@glo@text}%
3058 }%
3059 }%
3060 }%

```

Indicate that this entry has now been used

```

3061 \ifKV@glslink@local
3062 \glslocalunset{#2}%
3063 \else
3064 \glsunset{#2}%
3065 \fi
3066 }%
3067 }

```

`\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`

```
3068 \newrobustcmd*{\Glspl}{\@ifstar\@sGlspl\@Glspl}
```

Define the starred form:

```
3069 \newcommand*{\@sGlspl}[1][\@Glspl[hyper=false,#1]]
```

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

```
3070 \newcommand*{\@Glspl}[2][\@%
```

```
3071 \new@ifnextchar[\@Glspl@{#1}{#2}]{\@Glspl@{#1}{#2}[]}%
```

```
3072 }
```

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

```
3073 \def\@Glspl@#1#2[#3]{%
```

```
3074 \glsdoifexists{#2}%
```

```
3075 {%
```

```
3076 \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
3077 \def\@gls@link@opts{#1}%
```

```
3078 \def\@gls@link@label{#2}%
```

```
3079 \def\glslabel{#2}%
```

```
3080 \let\glsifplural\@firstoftwo
```

```
3081 \let\glschapscase\@secondofthree
```

```
3082 \let\glscustomtext\@empty
```

```
3083 \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`.

```
3084 \def\@glo@text{\csname gls@\@glo@type @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.

```
3085 \ifglsused{#2}%
```

```
3086 {%
```

```
3087 \@gls@link[#1]{#2}{\@glo@text}%
```

```
3088 }%
```

```
3089 {%
```

```
3090 \gls@checkisacronymlist\@glo@type
```

```
3091 \ifthenelse
```

```
3092 {%
```

```
3093 \(\boolean{\glsisacronymlist}\AND \boolean{\glsacrfootnote}\)
```

```
3094 \OR \NOT\boolean{\glshyperfirst}}%
```

```

3095     }%
3096     {%
3097     \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
3098     }%
3099     {%
3100     \@gls@link[#1]{#2}{\@glo@text}%
3101     }%
3102     }%

```

Indicate that this entry has now been used

```

3103     \ifKV@glslink@local
3104     \glslocalunset{#2}%
3105     \else
3106     \glsunset{#2}%
3107     \fi
3108     }%
3109 }

```

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

`\GLSp1`

```

3110 \newrobustcmd*{\GLSp1}{\@ifstar\@sGLSp1\@GLSp1}

```

Define the starred form:

```

3111 \newcommand*{\@sGLSp1}[1][\@GLSp1[hyper=false,#1]]

```

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

```

3112 \newcommand*{\@GLSp1}[2][\@GLSp1@{#1}{#2}]{\@GLSp1@{#1}{#2}[]}
3113 \new@ifnextchar[\@GLSp1@{#1}{#2}]{\@GLSp1@{#1}{#2}[]}
3114 }

```

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

```

3115 \def\@GLSp1@#1#2[#3]{%
3116   \glsdoifexists{#2}%
3117   {%
3118     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

3119   \def\@gls@link@opts{#1}%
3120   \def\@gls@link@label{#2}%

3121   \def\glslabel{#2}%
3122   \let\glsifplural\@firstoftwo
3123   \let\glsapscase\@thirdofthree
3124   \let\glscustomtext\@empty
3125   \def\glsinsert{#3}%

```

Determine what the link text should be (this is stored in `\@glo@text`)

```

3126   \def\@glo@text{\csname gls@\@glo@type @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 `hyper-first=false` package option is used.

```

3127 \ifglsused{#2}%
3128 {%
3129 \@gls@link[#1]{#2}{\@glo@text}%
3130 }%
3131 {%
3132 \gls@checkisacronymlist\@glo@type
3133 \ifthenelse
3134 {%
3135 \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
3136 \OR \NOT\boolean{glshyperfirst}}%
3137 }%
3138 {%
3139 \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
3140 }%
3141 {%
3142 \@gls@link[#1]{#2}{\@glo@text}%
3143 }%
3144 }%

```

Indicate that this entry has now been used

```

3145 \ifKV@glslink@local
3146 \glslocalunset{#2}%
3147 \else
3148 \glsunset{#2}%
3149 \fi
3150 }%
3151 }

```

`\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:

```

3152 \newrobustcmd*{\glsdisp}{\@ifstar\sglsdisp\@glsdisp}

```

Define the starred form:

`\@sgls`

```

3153 \newcommand*{\@sglsdisp}[1][\@glsdisp[hyper=false,#1]]{

```

Defined the un-starred form.

`\@glsdisp`

```

3154 \newcommand*{\@glsdisp}[3][\@glsdisp[hyper=false,#1]]{
3155 \glsdoifexists{#2}{%
3156 \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

3157 \def\@gls@link@opts{#1}%
3158 \def\@gls@link@label{#2}%

3159 \def\glslabel{#2}%
3160 \let\glsifplural\@secondoftwo
3161 \let\glsupcase\@firstofthree
3162 \def\glsustomtext{#3}%
3163 \def\glsinsert{}%

```

Determine what the link text should be (this is stored in \@glo@text)

```

3164 \def\@glo@text{\csname gls@\@glo@type @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 hyper-first=false package option is used.

```

3165 \ifglsused{#2}%
3166 {%
3167   \@gls@link[#1]{#2}{\@glo@text}%
3168 }%
3169 {%
3170   \gls@checkisacronymlist\@glo@type
3171   \ifthenelse{\boolean{@glsisacronymlist}\AND
3172     \boolean{glsacrfootnote}}{\OR \NOT\boolean{gls hyperfirst}}}%
3173 {%
3174   \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
3175 }%
3176 {%
3177   \@gls@link[#1]{#2}{\@glo@text}%
3178 }%
3179 }%

```

Indicate that this entry has now been used

```

3180 \ifKV@glslink@local
3181   \glslocalunset{#2}%
3182 \else
3183   \glsunset{#2}%
3184 \fi
3185 }%
3186 }

```

\gls{text} behaves like \gls except it always uses the value given by the text key and it doesn't mark the entry as used.

\gls{text}

```

3187 \newrobustcmd*{\gls{text}}{\@ifstar\@sgls{text}\@gls{text}}

```

Define the starred form:

```

3188 \newcommand*{\@sgls{text}}[1][\@gls{text}[hyper=false,#1]}

```

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

```
3189 \newcommand*{\@glstext}[2] [] {%
3190   \new@ifnextchar[{\@glstext@{#1}{#2}}{\@glstext@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3191 \def\@glstext@#1#2[#3] {%
3192   \glsdoifexists{#2}%
3193   {%
3194     \edef\@glo@type{\glstentrytype{#2}}%
```

Call \@gls@link

```
3195     \@gls@link[#1]{#2}{\glstentrytext{#2}#3}%
3196   }%
3197 }
```

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

\GLStext

```
3198 \newrobustcmd*{\GLStext}{\@ifstar\@sGLStext\@GLStext}
```

Define the starred form:

```
3199 \newcommand*{\@sGLStext}[1] [] {\@GLStext[hyper=false,#1]}
```

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

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

Read in the final optional argument:

```
3202 \def\@GLStext@#1#2[#3] {%
3203   \glsdoifexists{#2}%
3204   {%
3205     \edef\@glo@type{\glstentrytype{#2}}%
```

Call \@gls@link

```
3206     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glstentrytext{#2}#3}}%
3207   }%
3208 }
```

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

\Glstext

```
3209 \newrobustcmd*{\Glstext}{\@ifstar\@sGlstext\@Glstext}
```

Define the starred form:

```
3210 \newcommand*{\@sGlstext}[1] [] {\@Glstext[hyper=false,#1]}
```

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

```
3211 \newcommand*{\@Glstext}[2] [] {%
3212   \new@ifnextchar[{\@Glstext@{#1}{#2}}{\@Glstext@{#1}{#2} []}]}
```

Read in the final optional argument:

```

3213 \def\@Glstext@#1#2[#3]{%
3214   \glsdoifexists{#2}%
3215   {%
3216     \edef\@glo@type{\glsentrytype{#2}}%
3217     Call \@gls@link
3218     \@gls@link[#1]{#2}{\Glsentrytext{#2}#3}%
3219   }%
3220 }

```

\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

```

3220 \newrobustcmd*{\glsfirst}{\@ifstar\sglsfirst\@glsfirst}

```

Define the starred form:

```

3221 \newcommand*{\@sglsfirst}[1][\@glsfirst[hyper=false,#1]]

```

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

```

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

```

Read in the final optional argument:

```

3224 \def\@glsfirst@#1#2[#3]{%
3225   \glsdoifexists{#2}%
3226   {%
3227     \edef\@glo@type{\glsentrytype{#2}}%
3228     Call \@gls@link
3229     \@gls@link[#1]{#2}{\Glsentryfirst{#2}#3}%
3230   }%
3231 }

```

\Glsfirst behaves like \glsfirst except it displays the first letter in uppercase.

\Glsfirst

```

3231 \newrobustcmd*{\Glsfirst}{\@ifstar\@sGlsfirst\@Glsfirst}

```

Define the starred form:

```

3232 \newcommand*{\@sGlsfirst}[1][\@Glsfirst[hyper=false,#1]]

```

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

```

3233 \newcommand*{\@Glsfirst}[2][\@%
3234   \new@ifnextchar[\@Glsfirst@{#1}{#2}]{\@Glsfirst@{#1}{#2}[]}]

```

Read in the final optional argument:

```
3235 \def\@Glsfirst@#1#2[#3]{%
3236   \glsdoifexists{#2}%
3237   {%
3238     \edef\@glo@type{\glentrytype{#2}}%
```

Call \@gls@link

```
3239   \@gls@link[#1]{#2}{\Glsentryfirst{#2}#3}}%
3240 }
```

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

\GLSfirst

```
3241 \newrobustcmd*{\GLSfirst}{\@ifstar\@sGLSfirst\@GLSfirst}
```

Define the starred form:

```
3242 \newcommand*{\@sGLSfirst}[1][\@GLSfirst[hyper=false,#1]]
```

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

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

Read in the final optional argument:

```
3245 \def\@GLSfirst@#1#2[#3]{%
3246   \glsdoifexists{#2}%
3247   {%
3248     \edef\@glo@type{\glentrytype{#2}}%
```

Determine what the link text should be (this is stored in Call \@gls@link

```
3249   \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glentryfirst{#2}#3}}%
3250   }%
3251 }
```

\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

```
3252 \newrobustcmd*{\glsplural}{\@ifstar\@sglsplural\@glsplural}
```

Define the starred form:

```
3253 \newcommand*{\@sglsplural}[1][\@glsplural[hyper=false,#1]]
```

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

```
3254 \newcommand*{\@glsplural}[2][{%
3255   \new@ifnextchar[\@glsplural@{#1}{#2}]{\@glsplural@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3256 \def\@glsplural@#1#2[#3]{%
3257   \glsdoifexists{#2}%
3258   {%
3259     \edef\@glo@type{\glentrytype{#2}}%
```

Call \@gls@link

```

3260   \@gls@link[#1]{#2}{\glsentryplural{#2}#3}%
3261   }%
3262 }

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

```

\Glsplural

```

3263 \newrobustcmd*{\Glsplural}{\@ifstar\@sGlsplural\@Glsplural}

Define the starred form:
3264 \newcommand*{\@sGlsplural}[1] [] {\@Glsplural[hyper=false,#1]}

Defined the un-starred form. Need to determine if there is a final optional ar-
gument
3265 \newcommand*{\@Glsplural}[2] [] {%
3266   \new@ifnextchar[{\@Glsplural@{#1}{#2}}{\@Glsplural@{#1}{#2} []}]

Read in the final optional argument:
3267 \def\@Glsplural@#1#2[#3] {%
3268   \glsdoifexists{#2}%
3269   {%
3270     \edef\@glo@type{\glsentrytype{#2}}%

Call \@gls@link
3271   \@gls@link[#1]{#2}{\glsentryplural{#2}#3}%
3272   }%
3273 }

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

```

\GLSplural

```

3274 \newrobustcmd*{\GLSplural}{\@ifstar\@sGLSplural\@GLSplural}

Define the starred form:
3275 \newcommand*{\@sGLSplural}[1] [] {\@GLSplural[hyper=false,#1]}

Defined the un-starred form. Need to determine if there is a final optional ar-
gument
3276 \newcommand*{\@GLSplural}[2] [] {%
3277   \new@ifnextchar[{\@GLSplural@{#1}{#2}}{\@GLSplural@{#1}{#2} []}]

Read in the final optional argument:
3278 \def\@GLSplural@#1#2[#3] {%
3279   \glsdoifexists{#2}%
3280   {%
3281     \edef\@glo@type{\glsentrytype{#2}}%

Call \@gls@link
3282   \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentryplural{#2}#3}}%
3283   }%
3284 }

```

`\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`

```
3285 \newrobustcmd*{\glsfirstplural}{\@ifstar\sglsfirstplural\glsfirstplural}
```

Define the starred form:

```
3286 \newcommand*{\sglsfirstplural}[1] [] {\glsfirstplural[hyper=false,#1]}
```

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

```
3287 \newcommand*{\glsfirstplural}[2] [] {%
```

```
3288   \new@ifnextchar[{\glsfirstplural@{#1}{#2}}{\glsfirstplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3289 \def\glsfirstplural@#1#2[#3] {%
```

```
3290   \glsdoifexists{#2}%
```

```
3291   {%
```

```
3292     \edef\glo@type{\glsentrytype{#2}}%
```

Call `\@gls@link`

```
3293     \@gls@link[#1]{#2}{\glsentryfirstplural{#2}#3}%
```

```
3294   }%
```

```
3295 }
```

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

`\Glsfirstplural`

```
3296 \newrobustcmd*{\Glsfirstplural}{\@ifstar\sglsfirstplural\Glsfirstplural}
```

Define the starred form:

```
3297 \newcommand*{\sglsfirstplural}[1] [] {\Glsfirstplural[hyper=false,#1]}
```

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

```
3298 \newcommand*{\Glsfirstplural}[2] [] {%
```

```
3299   \new@ifnextchar[{\Glsfirstplural@{#1}{#2}}{\Glsfirstplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3300 \def\Glsfirstplural@#1#2[#3] {%
```

```
3301   \glsdoifexists{#2}%
```

```
3302   {%
```

```
3303     \edef\glo@type{\glsentrytype{#2}}%
```

Call `\@gls@link`

```
3304     \@gls@link[#1]{#2}{\Glsentryfirstplural{#2}#3}%
```

```
3305   }%
```

```
3306 }
```

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

`\GLSfirstplural`

```
3307 \newrobustcmd*{\GLSfirstplural}{\@ifstar\@sGLSfirstplural\@GLSfirstplural}
```

Define the starred form:

```
3308 \newcommand*{\@sGLSfirstplural}[1] [] {\@GLSfirstplural[hyper=false,#1]}
```

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

```
3309 \newcommand*{\@GLSfirstplural}[2] [] {%
```

```
3310   \new@ifnextchar[{\@GLSfirstplural@{#1}{#2}}{\@GLSfirstplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3311 \def\@GLSfirstplural@#1#2[#3] {%
```

```
3312   \glsdoifexists{#2}%
```

```
3313   {%
```

```
3314     \edef\@gls@type{\glsentrytype{#2}}%
```

Call `\@gls@link`

```
3315     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentryfirstplural{#2}#3}}%
```

```
3316   }%
```

```
3317 }
```

`\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`

```
3318 \newrobustcmd*{\glsname}{\@ifstar\@sglsname\@glsname}
```

Define the starred form:

```
3319 \newcommand*{\@sglsname}[1] [] {\@glsname[hyper=false,#1]}
```

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

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

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

Read in the final optional argument:

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

```
3323   \glsdoifexists{#2}%
```

```
3324   {%
```

```
3325     \edef\@gls@type{\glsentrytype{#2}}%
```

Call `\@gls@link`

```
3326     \@gls@link[#1]{#2}{\glsentryname{#2}#3}}%
```

```
3327   }%
```

```
3328 }
```

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

`\Glsname`

```
3329 \newrobustcmd*{\Glsname}{\@ifstar\@sGlsname\@Glsname}
```

Define the starred form:

```
3330 \newcommand*{\@sGlsname}[1] [] {\@Glsname[hyper=false,#1]}
```

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

```
3331 \newcommand*{\@Glsname}[2] [] {%
```

```
3332   \new@ifnextchar[\@Glsname@{#1}{#2}]{\@Glsname@{#1}{#2} []}}
```

Read in the final optional argument:

```
3333 \def\@Glsname@#1#2[#3] {%
```

```
3334   \glsdoifexists{#2}%
```

```
3335   {%
```

```
3336     \edef\@glo@type{\glsentrytype{#2}}%
```

Call \@gls@link

```
3337     \@gls@link[#1]{#2}{\glsentryname{#2}#3}%
```

```
3338   }%
```

```
3339 }
```

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

\GLSname

```
3340 \newrobustcmd*{\GLSname}{\@ifstar\@sGLSname\@GLSname}
```

Define the starred form:

```
3341 \newcommand*{\@sGLSname}[1] [] {\@GLSname[hyper=false,#1]}
```

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

```
3342 \newcommand*{\@GLSname}[2] [] {%
```

```
3343   \new@ifnextchar[\@GLSname@{#1}{#2}]{\@GLSname@{#1}{#2} []}}
```

Read in the final optional argument:

```
3344 \def\@GLSname@#1#2[#3] {%
```

```
3345   \glsdoifexists{#2}%
```

```
3346   {%
```

```
3347     \edef\@glo@type{\glsentrytype{#2}}%
```

Call \@gls@link

```
3348     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentryname{#2}#3}}%
```

```
3349   }%
```

```
3350 }
```

\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

```
3351 \newrobustcmd*{\glsdesc}{\@ifstar\@sglsdesc\@glsdesc}
```

Define the starred form:

```
3352 \newcommand*{\@sglsdesc}[1] [] {\@glsdesc[hyper=false,#1]}
```

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

```
3353 \newcommand*{\@glsdesc}[2][\%
3354 \new@ifnextchar[\@glsdesc@{#1}{#2}]{\@glsdesc@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3355 \def\@glsdesc@#1#2[#3]{\%
3356 \glsdoifexists{#2}\%
3357 {\%
3358 \edef\@glo@type{\glsentrytype{#2}}\%
```

Call \@gls@link

```
3359 \@gls@link[#1]{#2}{\glsentrydesc{#2}#3}\%
3360 }%
3361 }
```

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

\Glsdesc

```
3362 \newrobustcmd*{\Glsdesc}{\@ifstar\@sGlsdesc\@Glsdesc}
```

Define the starred form:

```
3363 \newcommand*{\@sGlsdesc}[1][\@Glsdesc[hyper=false,#1]]
```

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

```
3364 \newcommand*{\@Glsdesc}[2][\%
3365 \new@ifnextchar[\@Glsdesc@{#1}{#2}]{\@Glsdesc@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3366 \def\@Glsdesc@#1#2[#3]{\%
3367 \glsdoifexists{#2}\%
3368 {\%
3369 \edef\@glo@type{\glsentrytype{#2}}\%
```

Call \@gls@link

```
3370 \@gls@link[#1]{#2}{\Glsentrydesc{#2}#3}\%
3371 }%
3372 }
```

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

\GLSdesc

```
3373 \newrobustcmd*{\GLSdesc}{\@ifstar\@sGLSdesc\@GLSdesc}
```

Define the starred form:

```
3374 \newcommand*{\@sGLSdesc}[1][\@GLSdesc[hyper=false,#1]]
```

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

```
3375 \newcommand*{\@GLSdesc}[2][\%
3376 \new@ifnextchar[\@GLSdesc@{#1}{#2}]{\@GLSdesc@{#1}{#2}[]}}
```

Read in the final optional argument:

```

3377 \def\@GLSdesc@#1#2[#3]{%
3378   \glsdoifexists{#2}%
3379   {%
3380     \edef\@glo@type{\glsentrytype{#2}}%
3381     Call \@gls@link
3382     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentrydesc{#2}#3}}%
3383   }%
3384 }

```

`\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`

```

3384 \newrobustcmd*{\glsdescplural}{\@ifstar\@sglsdescplural\@glsdescplural}

```

Define the starred form:

```

3385 \newcommand*{\@sglsdescplural}[1][\@glsdescplural[hyper=false,#1]]{

```

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

```

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

```

Read in the final optional argument:

```

3388 \def\@glsdescplural@#1#2[#3]{%
3389   \glsdoifexists{#2}%
3390   {%
3391     \edef\@glo@type{\glsentrytype{#2}}%
3392     Call \@gls@link
3393     \@gls@link[#1]{#2}{\glsentrydescplural{#2}#3}%
3394   }%
3395 }

```

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

`\Glsdescplural`

```

3395 \newrobustcmd*{\Glsdescplural}{\@ifstar\@sGlsdescplural\@Glsdescplural}

```

Define the starred form:

```

3396 \newcommand*{\@sGlsdescplural}[1][\@Glsdescplural[hyper=false,#1]]{

```

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

```

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

```

Read in the final optional argument:

```
3399 \def\@GLSdescplural@#1#2[#3]{%
3400   \glsdoifexists{#2}%
3401   {%
3402     \edef\@glo@type{\glentrytype{#2}}%
```

Call \@gls@link

```
3403     \@gls@link[#1]{#2}{\glentrydescplural{#2}#3}%
3404   }%
3405 }
```

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

\GLSdescplural

```
3406 \newrobustcmd*{\GLSdescplural}{\@ifstar\@sGLSdescplural\@GLSdescplural}
```

Define the starred form:

```
3407 \newcommand*{\@sGLSdescplural}[1][\@GLSdescplural[hyper=false,#1]]
```

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

```
3408 \newcommand*{\@GLSdescplural}[2][\@GLSdescplural@{#1}{#2}[]]{%
3409   \new@ifnextchar{\@GLSdescplural@{#1}{#2}}{\@GLSdescplural@{#1}{#2}[]]}
```

Read in the final optional argument:

```
3410 \def\@GLSdescplural@#1#2[#3]{%
3411   \glsdoifexists{#2}%
3412   {%
3413     \edef\@glo@type{\glentrytype{#2}}%
```

Call \@gls@link

```
3414     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glentrydescplural{#2}#3}}%
3415   }%
3416 }
```

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

\glssymbol

```
3417 \newrobustcmd*{\glssymbol}{\@ifstar\@sglssymbol\@glssymbol}
```

Define the starred form:

```
3418 \newcommand*{\@sglssymbol}[1][\@glssymbol[hyper=false,#1]]
```

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

```
3419 \newcommand*{\@glssymbol}[2][\@glssymbol@{#1}{#2}[]]{%
3420   \new@ifnextchar{\@glssymbol@{#1}{#2}}{\@glssymbol@{#1}{#2}[]]}
```

Read in the final optional argument:

```
3421 \def\@glssymbol@#1#2[#3]{%
3422   \glsdoifexists{#2}%
3423   {%
3424     \edef\@glo@type{\glentrytype{#2}}%
```

Call \@gls@link

```
3425     \@gls@link[#1]{#2}{\glentrysymbol{#2}#3}%
3426   }%
3427 }
```

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

\Glsymbol

```
3428 \newrobustcmd*{\Glsymbol}{\@ifstar\@sGlsymbol\@Glsymbol}
```

Define the starred form:

```
3429 \newcommand*{\@sGlsymbol}[1][]{\@Glsymbol[hyper=false,#1]}
```

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

```
3430 \newcommand*{\@Glsymbol}[2][]{%
3431   \new@ifnextchar[{\@Glsymbol@{#1}{#2}}{\@Glsymbol@{#1}{#2}}{}}%
```

Read in the final optional argument:

```
3432 \def\@Glsymbol@#1#2[#3]{%
3433   \glsdoifexists{#2}%
3434   {%
3435     \edef\@glo@type{\glentrytype{#2}}%
```

Call \@gls@link

```
3436     \@gls@link[#1]{#2}{\glentrysymbol{#2}#3}%
3437   }%
3438 }
```

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

\GLSsymbol

```
3439 \newrobustcmd*{\GLSsymbol}{\@ifstar\@sGLSsymbol\@GLSsymbol}
```

Define the starred form:

```
3440 \newcommand*{\@sGLSsymbol}[1][]{\@GLSsymbol[hyper=false,#1]}
```

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

```
3441 \newcommand*{\@GLSsymbol}[2][]{%
3442   \new@ifnextchar[{\@GLSsymbol@{#1}{#2}}{\@GLSsymbol@{#1}{#2}}{}}%
```

Read in the final optional argument:

```

3443 \def\@GLSsymbol@#1#2[#3]{%
3444   \glsdoifexists{#2}%
3445   {%
3446     \edef\@glo@type{\glstrytype{#2}}%
3447     \@gls@link
3448   }%
3449 }
```

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

`\glsymbolplural`

```

3450 \newrobustcmd*{\glsymbolplural}{\@ifstar\@sglsymbolplural\glsymbolplural}
```

Define the starred form:

```

3451 \newcommand*{\@sglsymbolplural}[1][\@glsymbolplural[hyper=false,#1]]
```

Defined 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   \glsdoifexists{#2}%
3456   {%
3457     \edef\@glo@type{\glstrytype{#2}}%
3458     \@gls@link
3459   }%
3460 }
```

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

`\Glsymbolplural`

```

3461 \newrobustcmd*{\Glsymbolplural}{\@ifstar\@sGlsymbolplural\Glsymbolplural}
```

Define the starred form:

```

3462 \newcommand*{\@sGlsymbolplural}[1][\@Glsymbolplural[hyper=false,#1]]
```

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

```

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

Read in the final optional argument:

```
3465 \def\@Glsymbolplural@#1#2[#3]{%
3466   \glsdoifexists{#2}%
3467   {%
3468     \edef\@glo@type{\glentrytype{#2}}%
3469     \@gls@link[#1]{#2}{\Glsentrysymbolplural{#2}#3}%
3470   }%
3471 }
```

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

`\GLSsymbolplural`

```
3472 \newrobustcmd*{\GLSsymbolplural}{\@ifstar\@sGLSsymbolplural\@GLSsymbolplural}
```

Define the starred form:

```
3473 \newcommand*{\@sGLSsymbolplural}[1][\@GLSsymbolplural[hyper=false,#1]]
```

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

```
3474 \newcommand*{\@GLSsymbolplural}[2][\@GLSsymbolplural@#1]{%
3475   \new@ifnextchar[\@GLSsymbolplural@{#1}{#2}]{\@GLSsymbolplural@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3476 \def\@GLSsymbolplural@#1#2[#3]{%
3477   \glsdoifexists{#2}%
3478   {%
3479     \edef\@glo@type{\glentrytype{#2}}%
3480     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glentrysymbolplural{#2}#3}}%
3481   }%
3482 }
```

`\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`

```
3483 \newrobustcmd*{\glsuseri}{\@ifstar\@sglsuseri\@glsuseri}
```

Define the starred form:

```
3484 \newcommand*{\@sglsuseri}[1][\@glsuseri[hyper=false,#1]]
```

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

```
3485 \newcommand*{\@glsuseri}[2][\@glsuseri@#1]{%
3486   \new@ifnextchar[\@glsuseri@{#1}{#2}]{\@glsuseri@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3487 \def\@glsuseri@#1#2[#3]{%
3488   \glsdoifexists{#2}%
3489   {%
3490     \edef\@glo@type{\glentrytype{#2}}%
```

Call \@gls@link

```

3491 \@gls@link[#1]{#2}{\glsentryuseri{#2}#3}%
3492 }%
3493 }

```

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

\Glsuseri

```

3494 \newrobustcmd*{\Glsuseri}{\@ifstar\@sGlsuseri\@Glsuseri}

```

Define the starred form:

```

3495 \newcommand*{\@sGlsuseri}[1][\@Glsuseri[hyper=false,#1]]{

```

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

```

3496 \newcommand*{\@Glsuseri}[2][\@Glsuseri@{#1}{#2}]{\@Glsuseri@{#1}{#2}[]}
3497 \new@ifnextchar[\@Glsuseri@{#1}{#2}]{\@Glsuseri@{#1}{#2}[]}

```

Read in the final optional argument:

```

3498 \def\@Glsuseri@#1#2[#3]{%
3499 \glsdoifexists{#2}%
3500 {%
3501 \edef\@glo@type{\glsentrytype{#2}}%

```

Call \@gls@link

```

3502 \@gls@link[#1]{#2}{\glsentryuseri{#2}#3}%
3503 }%
3504 }

```

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

\GLSuseri

```

3505 \newrobustcmd*{\GLSuseri}{\@ifstar\@sGLSuseri\@GLSuseri}

```

Define the starred form:

```

3506 \newcommand*{\@sGLSuseri}[1][\@GLSuseri[hyper=false,#1]]{

```

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

```

3507 \newcommand*{\@GLSuseri}[2][\@GLSuseri@{#1}{#2}]{\@GLSuseri@{#1}{#2}[]}
3508 \new@ifnextchar[\@GLSuseri@{#1}{#2}]{\@GLSuseri@{#1}{#2}[]}

```

Read in the final optional argument:

```

3509 \def\@GLSuseri@#1#2[#3]{%
3510 \glsdoifexists{#2}%
3511 {%
3512 \edef\@glo@type{\glsentrytype{#2}}%

```

Call \@gls@link

```

3513 \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentryuseri{#2}#3}}%
3514 }%
3515 }

```

`\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`

```
3516 \newrobustcmd*{\glsuserii}{\@ifstar\sglsuserii\@glsuserii}
```

Define the starred form:

```
3517 \newcommand*{\sglsuserii}[1] [] {\@glsuserii[hyper=false,#1]}
```

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

```
3518 \newcommand*{\@glsuserii}[2] [] {%
```

```
3519   \new@ifnextchar[{\@glsuserii@{#1}{#2}}{\@glsuserii@{#1}{#2} []}]
```

Read in the final optional argument:

```
3520 \def\@glsuserii@#1#2[#3] {%
```

```
3521   \glsdoifexists{#2}%
```

```
3522   {%
```

```
3523     \edef\@gls@type{\glsentrytype{#2}}%
```

Call `\@gls@link`

```
3524     \@gls@link[#1]{#2}{\glsentryuserii{#2}#3}%
```

```
3525   }%
```

```
3526 }
```

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

`\Glsuserii`

```
3527 \newrobustcmd*{\Glsuserii}{\@ifstar\@sGlsuserii\@Glsuserii}
```

Define the starred form:

```
3528 \newcommand*{\@sGlsuserii}[1] [] {\@Glsuserii[hyper=false,#1]}
```

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

```
3529 \newcommand*{\@Glsuserii}[2] [] {%
```

```
3530   \new@ifnextchar[{\@Glsuserii@{#1}{#2}}{\@Glsuserii@{#1}{#2} []}]
```

Read in the final optional argument:

```
3531 \def\@Glsuserii@#1#2[#3] {%
```

```
3532   \glsdoifexists{#2}%
```

```
3533   {%
```

```
3534     \edef\@gls@type{\glsentrytype{#2}}%
```

Call `\@gls@link`

```
3535     \@gls@link[#1]{#2}{\Glsentryuserii{#2}#3}%
```

```
3536   }%
```

```
3537 }
```

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

`\GLSuserii`

```
3538 \newrobustcmd*{\GLSuserii}{\@ifstar\@sGLSuserii\@GLSuserii}
```

Define the starred form:

```
3539 \newcommand*{\@sGLSuserii}[1] [] {\@GLSuserii[hyper=false,#1]}
```

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

```
3540 \newcommand*{\@GLSuserii}[2] [] {%
```

```
3541   \new@ifnextchar[{\@GLSuserii@{#1}{#2}}{\@GLSuserii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3542 \def\@GLSuserii@#1#2[#3] {%
```

```
3543   \glsdoifexists{#2}%
```

```
3544   {%
```

```
3545     \edef\@gls@type{\glsentrytype{#2}}%
```

Call `\@gls@link`

```
3546     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentryuserii{#2}#3}}%
```

```
3547   }%
```

```
3548 }
```

`\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`

```
3549 \newrobustcmd*{\glsuseriii}{\@ifstar\@sglsuseriii\@glsuseriii}
```

Define the starred form:

```
3550 \newcommand*{\@sglsuseriii}[1] [] {\@glsuseriii[hyper=false,#1]}
```

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

```
3551 \newcommand*{\@glsuseriii}[2] [] {%
```

```
3552   \new@ifnextchar[{\@glsuseriii@{#1}{#2}}{\@glsuseriii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3553 \def\@glsuseriii@#1#2[#3] {%
```

```
3554   \glsdoifexists{#2}%
```

```
3555   {%
```

```
3556     \edef\@gls@type{\glsentrytype{#2}}%
```

Call `\@gls@link`

```
3557     \@gls@link[#1]{#2}{\glsentryuseriii{#2}#3}%
```

```
3558   }%
```

```
3559 }
```

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

`\Glsuseriii`

```
3560 \newrobustcmd*{\Glsuseriii}{\@ifstar\@sGlsuseriii\@Glsuseriii}
```

Define the starred form:

```
3561 \newcommand*{\@sGlsuseriii}[1] [] {\@Glsuseriii[hyper=false,#1]}
```

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

```
3562 \newcommand*{\@Glsuseriii}[2] [] {%
```

```
3563   \new@ifnextchar[{\@Glsuseriii@{#1}{#2}}]{\@Glsuseriii@{#1}{#2} []}}
```

Read in the final optional argument:

```
3564 \def\@Glsuseriii@#1#2[#3]{%
```

```
3565   \glsdoifexists{#2}%
```

```
3566   {%
```

```
3567     \edef\@gls@type{\glsentrytype{#2}}%
```

Call \@gls@link

```
3568     \@gls@link[#1]{#2}{\glsentryuseriii{#2}#3}%
```

```
3569   }%
```

```
3570 }
```

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

\GLSuseriii

```
3571 \newrobustcmd*{\GLSuseriii}{\@ifstar\@sGLSuseriii\@GLSuseriii}
```

Define the starred form:

```
3572 \newcommand*{\@sGLSuseriii}[1] [] {\@GLSuseriii[hyper=false,#1]}
```

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

```
3573 \newcommand*{\@GLSuseriii}[2] [] {%
```

```
3574   \new@ifnextchar[{\@GLSuseriii@{#1}{#2}}]{\@GLSuseriii@{#1}{#2} []}}
```

Read in the final optional argument:

```
3575 \def\@GLSuseriii@#1#2[#3]{%
```

```
3576   \glsdoifexists{#2}%
```

```
3577   {%
```

```
3578     \edef\@gls@type{\glsentrytype{#2}}%
```

Call \@gls@link

```
3579     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentryuseriii{#2}#3}}%
```

```
3580   }%
```

```
3581 }
```

\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

```
3582 \newrobustcmd*{\glsuseriv}{\@ifstar\@sglsuseriv\@glsuseriv}
```

Define the starred form:

```
3583 \newcommand*{\@sglsuseriv}[1] [] {\@glsuseriv[hyper=false,#1]}
```

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

```
3584 \newcommand*{\@glsuseriv}[2] [] {%
3585   \new@ifnextchar[{\@glsuseriv@{#1}{#2}}{\@glsuseriv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3586 \def\@glsuseriv@#1#2[#3] {%
3587   \glsdoifexists{#2}%
3588   {%
3589     \edef\@glo@type{\glsentrytype{#2}}%
3590 % Call \cs{@gls@link}
3591 % \changes{3.11a}{2013-10-15}{changed to just use \cs{glsentryuseriv}}
3592 %   \begin{macrocode}
3593   \@gls@link[#1]{#2}{\glsentryuseriv{#2}#3}%
3594 }%
3595 }
```

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

`\Glsuseriv`

```
3596 \newrobustcmd*{\Glsuseriv}{\@ifstar\@sGlsuseriv\@Glsuseriv}
```

Define the starred form:

```
3597 \newcommand*{\@sGlsuseriv}[1] [] {\@Glsuseriv[hyper=false,#1]}
```

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

```
3598 \newcommand*{\@Glsuseriv}[2] [] {%
3599   \new@ifnextchar[{\@Glsuseriv@{#1}{#2}}{\@Glsuseriv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3600 \def\@Glsuseriv@#1#2[#3] {%
3601   \glsdoifexists{#2}%
3602   {%
3603     \edef\@glo@type{\glsentrytype{#2}}%
3604     \@gls@link[#1]{#2}{\Glsentryuseriv{#2}#3}%
3605   }%
3606 }
```

Call `\@gls@link`

```
3604   \@gls@link[#1]{#2}{\Glsentryuseriv{#2}#3}%
3605 }%
3606 }
```

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

`\GLSuseriv`

```
3607 \newrobustcmd*{\GLSuseriv}{\@ifstar\@sGLSuseriv\@GLSuseriv}
```

Define the starred form:

```
3608 \newcommand*{\@sGLSuseriv}[1] [] {\@GLSuseriv[hyper=false,#1]}
```

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

```
3609 \newcommand*{\@GLSuseriv}[2] [] {%
3610   \new@ifnextchar[{\@GLSuseriv@{#1}{#2}}{\@GLSuseriv@{#1}{#2} [] }}
```

Read in the final optional argument:

```
3611 \def\@GLSuseriv@#1#2[#3] {%
3612   \glsdoifexists{#2}%
3613   {%
3614     \edef\@gls@type{\glsentrytype{#2}}%
```

Call \@gls@link

```
3615     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentryuseriv{#2}#3}}%
3616   }%
3617 }
```

\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

```
3618 \newrobustcmd*{\glsuserv}{\@ifstar\@sglsuserv\@glsuserv}
```

Define the starred form:

```
3619 \newcommand*{\@sglsuserv}[1] [] {\@glsuserv[hyper=false,#1]}
```

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

```
3620 \newcommand*{\@glsuserv}[2] [] {%
3621   \new@ifnextchar[{\@glsuserv@{#1}{#2}}{\@glsuserv@{#1}{#2} [] }}
```

Read in the final optional argument:

```
3622 \def\@glsuserv@#1#2[#3] {%
3623   \glsdoifexists{#2}%
3624   {%
3625     \edef\@gls@type{\glsentrytype{#2}}%
```

Call \@gls@link

```
3626     \@gls@link[#1]{#2}{\glsentryuserv{#2}#3}%
3627   }%
3628 }
```

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

\Glsuserv

```
3629 \newrobustcmd*{\Glsuserv}{\@ifstar\@sGlsuserv\@Glsuserv}
```

Define the starred form:

```
3630 \newcommand*{\@sGlsuserv}[1] [] {\@Glsuserv[hyper=false,#1]}
```

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

```
3631 \newcommand*{\@Glsuserv}[2] [] {%
3632   \new@ifnextchar[{\@Glsuserv@{#1}{#2}}{\@Glsuserv@{#1}{#2} [] }}
```

Read in the final optional argument:

```
3633 \def\@Glsuserv@#1#2[#3]{%
3634   \glsdoifexists{#2}%
3635   {%
3636     \edef\@gls@type{\glsentrytype{#2}}%
3637     \@gls@link[#1]{#2}{\glsentryuserv{#2}#3}%
3638   }%
3639 }
```

Call \@gls@link

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

\Glsuserv

```
3640 \newrobustcmd*{\@Glsuserv}{\@ifstar\@sGlsuserv\@Glsuserv}
```

Define the starred form:

```
3641 \newcommand*{\@sGlsuserv}[1][\@Glsuserv[hyper=false,#1]]
```

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

```
3642 \newcommand*{\@Glsuserv}[2][\@Glsuserv@{#1}{#2}[]]{%
3643 \new@ifnextchar[\@Glsuserv@{#1}{#2}]{\@Glsuserv@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3644 \def\@Glsuserv@#1#2[#3]{%
3645   \glsdoifexists{#2}%
3646   {%
3647     \edef\@gls@type{\glsentrytype{#2}}%
3648     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glsentryuserv{#2}#3}}%
3649   }%
3650 }
```

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

\glsuservi

```
3651 \newrobustcmd*{\glsuservi}{\@ifstar\@sglsuservi\@glsuservi}
```

Define the starred form:

```
3652 \newcommand*{\@sglsuservi}[1][\@glsuservi[hyper=false,#1]]
```

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

```
3653 \newcommand*{\@glsuservi}[2][\@glsuservi@{#1}{#2}[]]{%
3654   \new@ifnextchar[\@glsuservi@{#1}{#2}]{\@glsuservi@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3655 \def\@glsuservi@#1#2[#3]{%
3656   \glsdoifexists{#2}%
3657   {%
3658     \edef\@glo@type{\glsentrytype{#2}}%
3659     Call \@gls@link
3659     \@gls@link[#1]{#2}{\glsentryuservi{#2}#3}%
3660   }%
3661 }
```

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

`\Glsuservi`

```
3662 \newrobustcmd*{\Glsuservi}{\@ifstar\@sGlsuservi\@Glsuservi}
```

Define the starred form:

```
3663 \newcommand*{\@sGlsuservi}[1][]{\@Glsuservi[hyper=false,#1]}
```

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

```
3664 \newcommand*{\@Glsuservi}[2][]{%
3665   \new@ifnextchar[\@Glsuservi@{#1}{#2}]{\@Glsuservi@{#1}{#2}[]}}
```

Read in the final optional argument:

```
3666 \def\@Glsuservi@#1#2[#3]{%
3667   \glsdoifexists{#2}%
3668   {%
3669     \edef\@glo@type{\glsentrytype{#2}}%
3670     Call \@gls@link
3670     \@gls@link[#1]{#2}{\Glsentryuservi{#2}#3}%
3671   }%
3672 }
```

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

`\GLSuservi`

```
3673 \newrobustcmd*{\GLSuservi}{\@ifstar\@sGLSuservi\@GLSuservi}
```

Define the starred form:

```
3674 \newcommand*{\@sGLSuservi}[1][]{\@GLSuservi[hyper=false,#1]}
```

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

```
3675 \newcommand*{\@GLSuservi}[2][]{%
3676   \new@ifnextchar[\@GLSuservi@{#1}{#2}]{\@GLSuservi@{#1}{#2}[]}}
```

Read in the final optional argument:

```

3677 \def\@GLSuservi@#1#2[#3]{%
3678   \glsdoifexists{#2}%
3679   {%
3680     \edef\@glo@type{\glentrytype{#2}}%

    Call \@gls@link
3681     \@gls@link[#1]{#2}{\mfirstucMakeUppercase{\glentryuservi{#2}#3}}%
3682   }%
3683 }
```

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

\acrshort

```

3684 \newrobustcmd*{\acrshort}{\@ifstar\s@acrshort\ns@acrshort}
```

Define the starred form:

```

3685 \newcommand*{\s@acrshort}[2][{}]{%
3686   \new@ifnextchar[{\@acrshort{hyper=false,#1}{#2}}%
3687   {\@acrshort{hyper=false,#1}{#2}[]}%
3688 }
```

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

```

3689 \newcommand*{\ns@acrshort}[2][{}]{%
3690   \new@ifnextchar[{\@acrshort{#1}{#2}}{\@acrshort{#1}{#2}[]}%
3691 }
```

Read in the final optional argument:

```

3692 \def\@acrshort#1#2[#3]{%
3693   \glsdoifexists{#2}%
3694   {%
3695     \edef\@glo@type{\glentrytype{#2}}%

3696     \def\glslabel{#2}%
3697     \let\glsifplural\@secondoftwo
3698     \let\glscapscase\@firstofthree
3699     \let\glsinsert\@empty
3700     \def\glscustomtext{%
3701       \acronymfont{\glentryshort{#2}}#3%
3702     }%
```

Call \@gls@link

```

3703     \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3704   }%
3705 }
```

\Acrshort

```

3706 \newrobustcmd*{\Acrshort}{\@ifstar\s@Acrshort\ns@Acrshort}
```

Define the starred form:

```
3707 \newcommand*{\s@Acrshort}[2] [] {%
3708   \new@ifnextchar[{\@Acrshort{hyper=false,#1}{#2}}]{%
3709     {\@Acrshort{hyper=false,#1}{#2} []}%
3710 }
```

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

```
3711 \newcommand*{\ns@Acrshort}[2] [] {%
3712   \new@ifnextchar[{\@Acrshort{#1}{#2}}{\@Acrshort{#1}{#2} []}%
3713 }
```

Read in the final optional argument:

```
3714 \def\@Acrshort#1#2[#3] {%
3715   \glsdoifexists{#2}%
3716   {%
3717     \edef\@glo@type{\glsentrytype{#2}}%

3718     \def\glslabel{#2}%
3719     \let\glsifplural\@secondoftwo
3720     \let\glsupcase\@secondofthree
3721     \let\glsinsert\@empty
3722     \def\glscustomtext{%
3723       \acronymfont{\Glsentryshort{#2}}#3%
3724     }%
```

Call \@gls@link

```
3725   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3726   }%
3727 }
```

\ACRshort

```
3728 \newrobustcmd*{\ACRshort}{\@ifstar\s@ACRshort\ns@ACRshort}
```

Define the starred form:

```
3729 \newcommand*{\s@ACRshort}[2] [] {%
3730   \new@ifnextchar[{\@ACRshort{hyper=false,#1}{#2}}]{%
3731     {\@ACRshort{hyper=false,#1}{#2} []}%
3732 }
```

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

```
3733 \newcommand*{\ns@ACRshort}[2] [] {%
3734   \new@ifnextchar[{\@ACRshort{#1}{#2}}{\@ACRshort{#1}{#2} []}%
3735 }
```

Read in the final optional argument:

```
3736 \def\@ACRshort#1#2[#3] {%
3737   \glsdoifexists{#2}%
3738   {%
3739     \edef\@glo@type{\glsentrytype{#2}}%
```

```

3740 \def\glslabel{#2}%
3741 \let\glsifplural\@secondoftwo
3742 \let\glscapscase\@thirdofthree
3743 \let\glsinsert\@empty
3744 \def\glscustomtext{%
3745     \mfirstucMakeUppercase{\acronymfont{\glsentryshort{#2}}#3}%
3746 }%

```

Call \@gls@link

```

3747 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3748 }%
3749 }

```

Short plural:

\acrshortpl

```

3750 \newrobustcmd*{\acrshortpl}{\ifstar\s@acrshortpl\ns@acrshortpl}

```

Define the starred form:

```

3751 \newcommand*{\s@acrshortpl}[2] [] {%
3752     \new@ifnextchar[{\acrshortpl{hyper=false,#1}{#2}}%
3753         {\acrshortpl{hyper=false,#1}{#2} []}%
3754 }

```

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

```

3755 \newcommand*{\ns@acrshortpl}[2] [] {%
3756     \new@ifnextchar[{\acrshortpl{#1}{#2}}{\acrshortpl{#1}{#2} []}%
3757 }

```

Read in the final optional argument:

```

3758 \def\@acrshortpl#1#2[#3] {%
3759     \glsdoifexists{#2}%
3760     {%
3761         \edef\@glo@type{\glsentrytype{#2}}%
3762         \def\glslabel{#2}%
3763         \let\glsifplural\@firstoftwo
3764         \let\glscapscase\@firstofthree
3765         \let\glsinsert\@empty
3766         \def\glscustomtext{%
3767             \acronymfont{\glsentryshortpl{#2}}#3%
3768         }%

```

Call \@gls@link

```

3769 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3770 }%
3771 }

```

\Acrshortpl

```

3772 \newrobustcmd*{\Acrshortpl}{\ifstar\s@Acrshortpl\ns@Acrshortpl}

```

Define the starred form:

```
3773 \newcommand*{\s@Acrshortpl}[2] [] {%
3774   \new@ifnextchar[{\@Acrshortpl{hyper=false,#1}{#2}}}%
3775   {\@Acrshortpl{hyper=false,#1}{#2} []}%
3776 }
```

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

```
3777 \newcommand*{\ns@Acrshortpl}[2] [] {%
3778   \new@ifnextchar[{\@Acrshortpl{#1}{#2}}{\@Acrshortpl{#1}{#2} []}%
3779 }
```

Read in the final optional argument:

```
3780 \def\@Acrshortpl#1#2[#3] {%
3781   \glsdoifexists{#2}%
3782   {%
3783     \edef\@glo@type{\glsentrytype{#2}}%

3784     \def\glslabel{#2}%
3785     \let\glsifplural\@firstoftwo
3786     \let\glsupcase\@secondofthree
3787     \let\glsinsert\@empty
3788     \def\glscustomtext{%
3789       \acronymfont{\Glsentryshortpl{#2}}#3%
3790     }%
```

Call \@gls@link

```
3791   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3792   }%
3793 }
```

\ACRshortpl

```
3794 \newrobustcmd*{\ACRshortpl}{\@ifstar\s@ACRshortpl\ns@ACRshortpl}
```

Define the starred form:

```
3795 \newcommand*{\s@ACRshortpl}[2] [] {%
3796   \new@ifnextchar[{\@ACRshortpl{hyper=false,#1}{#2}}}%
3797   {\@ACRshortpl{hyper=false,#1}{#2} []}%
3798 }
```

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

```
3799 \newcommand*{\ns@ACRshortpl}[2] [] {%
3800   \new@ifnextchar[{\@ACRshortpl{#1}{#2}}{\@ACRshortpl{#1}{#2} []}%
3801 }
```

Read in the final optional argument:

```
3802 \def\@ACRshortpl#1#2[#3] {%
3803   \glsdoifexists{#2}%
3804   {%
3805     \edef\@glo@type{\glsentrytype{#2}}%
```

```

3806 \def\glslabel{#2}%
3807 \let\glsifplural\@firstoftwo
3808 \let\glscapscase\@thirdofthree
3809 \let\glsinsert\@empty
3810 \def\glscustomtext{%
3811 \mfirstucMakeUppercase{\acronymfont{\glentryshortpl{#2}}#3}%
3812 }%

Call \@gls@link
3813 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3814 }%
3815 }

```

\acrlong

```

3816 \newrobustcmd*{\acrlong}{\@ifstar\s@acrlong\@ns@acrlong}

```

Define the starred form:

```

3817 \newcommand*{\s@acrlong}[2] [] {%
3818 \new@ifnextchar[{\@acrlong{hyper=false,#1}{#2}}%
3819 {\@acrlong{hyper=false,#1}{#2} []}%
3820 }

```

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

```

3821 \newcommand*{\ns@acrlong}[2] [] {%
3822 \new@ifnextchar[{\@acrlong{#1}{#2}}{\@acrlong{#1}{#2} []}%
3823 }

```

Read in the final optional argument:

```

3824 \def\@acrlong#1#2[#3] {%
3825 \glsdoifexists{#2}%
3826 {%
3827 \edef\@glo@type{\glentrytype{#2}}%

3828 \def\glslabel{#2}%
3829 \let\glsifplural\@secondoftwo
3830 \let\glscapscase\@firstofthree
3831 \let\glsinsert\@empty

```

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

```

3832 \def\glscustomtext{%
3833 \glentrylong{#2}#3%
3834 }%

```

Call \@gls@link

```

3835 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3836 }%
3837 }

```

\Acrlong

```
3838 \newrobustcmd*{\Acrlong}{\@ifstar\s@Acrlong\@ns@Acrlong}
```

Define the starred form:

```
3839 \newcommand*{\s@Acrlong}[2] [] {%
3840   \new@ifnextchar[{\@Acrlong{hyper=false,#1}{#2}}%
3841     {\@Acrlong{hyper=false,#1}{#2} []}%
3842 }
```

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

```
3843 \newcommand*{\ns@Acrlong}[2] [] {%
3844   \new@ifnextchar[{\@Acrlong{#1}{#2}}{\@Acrlong{#1}{#2} []}%
3845 }
```

Read in the final optional argument:

```
3846 \def\@Acrlong#1#2[#3] {%
3847   \glsdoifexists{#2}%
3848   {%
3849     \edef\@glo@type{\glsentrytype{#2}}%
3850     \def\glslabel{#2}%
3851     \let\glsifplural\@secondoftwo
3852     \let\glsapscase\@secondofthree
3853     \let\glsinsert\@empty
```

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

```
3854   \def\glscustomtext{%
3855     \Glsentrylong{#2}#3%
3856   }%
```

Call \@gls@link

```
3857   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3858   }%
3859 }
```

\ACRlong

```
3860 \newrobustcmd*{\ACRlong}{\@ifstar\s@ACRlong\@ns@ACRlong}
```

Define the starred form:

```
3861 \newcommand*{\s@ACRlong}[2] [] {%
3862   \new@ifnextchar[{\@ACRlong{hyper=false,#1}{#2}}%
3863     {\@ACRlong{hyper=false,#1}{#2} []}%
3864 }
```

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

```
3865 \newcommand*{\ns@ACRlong}[2] [] {%
3866   \new@ifnextchar[{\@ACRlong{#1}{#2}}{\@ACRlong{#1}{#2} []}%
3867 }
```

Read in the final optional argument:

```
3868 \def\@ACRlong#1#2[#3]{%
3869   \glsdoifexists{#2}%
3870   {%
3871     \edef\@glo@type{\glsentrytype{#2}}%

3872   \def\glslabel{#2}%
3873   \let\glsifplural\@secondoftwo
3874   \let\glscapscase\@thirdofthree
3875   \let\glsinsert\@empty
```

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

```
3876   \def\glscustomtext{%
3877     \mfirstucMakeUppercase{\glsentrylong{#2}#3}%
3878   }%
```

Call \@gls@link

```
3879   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3880   }%
3881 }
```

Short plural:

\acrlongpl

```
3882 \newrobustcmd*{\acrlongpl}{\@ifstar\s@acrlongpl\ns@acrlongpl}
```

Define the starred form:

```
3883 \newcommand*{\s@acrlongpl}[2] [] {%
3884   \new@ifnextchar[{\@acrlongpl{hyper=false,#1}{#2}}%
3885   {\@acrlongpl{hyper=false,#1}{#2} []}%
3886 }
```

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

```
3887 \newcommand*{\ns@acrlongpl}[2] [] {%
3888   \new@ifnextchar[{\@acrlongpl{#1}{#2}}{\@acrlongpl{#1}{#2} []}%
3889 }
```

Read in the final optional argument:

```
3890 \def\@acrlongpl#1#2[#3]{%
3891   \glsdoifexists{#2}%
3892   {%
3893     \edef\@glo@type{\glsentrytype{#2}}%

3894   \def\glslabel{#2}%
3895   \let\glsifplural\@firstoftwo
3896   \let\glscapscase\@firstofthree
3897   \let\glsinsert\@empty
```

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

```

3898 \def\glscustomtext{%
3899 \glentrylongpl{#2}#3%
3900 }%

Call \@gls@link
3901 \@gls@link[#1]{#2}{\csname gls@%glo@type @entryfmt\endcsname}%
3902 }%
3903 }

```

\Acrlongpl

```

3904 \newrobustcmd*{\Acrlongpl}{\@ifstar\s@Acrlongpl\ns@Acrlongpl}

```

Define the starred form:

```

3905 \newcommand*{\s@Acrlongpl}[2][]{%
3906 \new@ifnextchar[{\@Acrlongpl{hyper=false#1}{#2}}%
3907 {\@Acrlongpl{hyper=false,#1}{#2}[]}%
3908 }

```

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

```

3909 \newcommand*{\ns@Acrlongpl}[2][]{%
3910 \new@ifnextchar[{\@Acrlongpl{#1}{#2}}{\@Acrlongpl{#1}{#2}[]}%
3911 }

```

Read in the final optional argument:

```

3912 \def\@Acrlongpl#1#2[#3]{%
3913 \glsoifexists{#2}%
3914 {%
3915 \edef\@glo@type{\glentrytype{#2}}%

3916 \def\glslabel{#2}%
3917 \let\glsifplural\@firstoftwo
3918 \let\glscapscase\@secondofthree
3919 \let\glsinsert\@empty

```

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

```

3920 \def\glscustomtext{%
3921 \Glsentrylongpl{#2}#3%
3922 }%

Call \@gls@link
3923 \@gls@link[#1]{#2}{\csname gls@%glo@type @entryfmt\endcsname}%
3924 }%
3925 }

```

\ACRlongpl

```

3926 \newrobustcmd*{\ACRlongpl}{\@ifstar\s@ACRlongpl\ns@ACRlongpl}

```

Define the starred form:

```
3927 \newcommand*{\s@ACRlongpl}[2] [] {%
3928   \new@ifnextchar[{\@ACRlongpl{hyper=false,#1}{#2}}{%
3929     {\@ACRlongpl{hyper=false,#1}{#2} []}%
3930 }
```

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

```
3931 \newcommand*{\ns@ACRlongpl}[2] [] {%
3932   \new@ifnextchar[{\@ACRlongpl{#1}{#2}}{\@ACRlongpl{#1}{#2} []}%
3933 }
```

Read in the final optional argument:

```
3934 \def\@ACRlongpl#1#2[#3] {%
3935   \glsdoifexists{#2}%
3936   {%
3937     \edef\@glo@type{\glsentrytype{#2}}%

3938   \def\glslabel{#2}%
3939   \let\glsifplural\@firstoftwo
3940   \let\glscapscase\@thirdofthree
3941   \let\glsinsert\@empty
```

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

```
3942   \def\glscustomtext{%
3943     \mfirstucMakeUppercase{\glsentrylongpl{#2}#3}%
3944   }%
```

Call \@gls@link

```
3945   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3946   }%
3947 }
```

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.

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

```
3948 \newcommand*{\glsentryname}[1]{\csname glo@#1@name\endcsname}
```

\Glsentryname

```
3949 \newrobustcmd*{\Glsentryname}[1]{%
```

```

3950 \protected@edef\@glo@text{\csname glo@#1@name\endcsname}%
3951 \expandafter\makefirstuc\expandafter{\@glo@text}%
3952 }

```

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`

```

3953 \newcommand*{\glsentrydesc}[1]{\csname glo@#1@desc\endcsname}

```

`\Glsentrydesc`

```

3954 \newrobustcmd*{\Glsentrydesc}[1]{%
3955 \protected@edef\@glo@text{\csname glo@#1@desc\endcsname}%
3956 \expandafter\makefirstuc\expandafter{\@glo@text}%
3957 }

```

Plural form:

`\glsentrydescplural`

```

3958 \newcommand*{\glsentrydescplural}[1]{%
3959 \csname glo@#1@descplural\endcsname}

```

`\Glsentrydescplural`

```

3960 \newrobustcmd*{\Glsentrydescplural}[1]{%
3961 \protected@edef\@glo@text{\csname glo@#1@descplural\endcsname}%
3962 \expandafter\makefirstuc\expandafter{\@glo@text}}

```

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`

```

3963 \newcommand*{\glsentrytext}[1]{\csname glo@#1@text\endcsname}

```

`\Glsentrytext`

```

3964 \newrobustcmd*{\Glsentrytext}[1]{%
3965 \protected@edef\@glo@text{\csname glo@#1@text\endcsname}%
3966 \expandafter\makefirstuc\expandafter{\@glo@text}}

```

Get the plural form:

`\glsentryplural`

```

3967 \newcommand*{\glsentryplural}[1]{\csname glo@#1@plural\endcsname}

```

`\Glsentryplural`

```

3968 \newrobustcmd*{\Glsentryplural}[1]{%
3969 \protected@edef\@glo@text{\csname glo@#1@plural\endcsname}%
3970 \expandafter\makefirstuc\expandafter{\@glo@text}}

```

Get the symbol associated with this entry. The argument is the label associated with the entry. Note that unless you used `symbol=false` in the `sanitize` package option you may get unexpected results if the symbol key contained any commands.

`\glsentrysymbol`

```
3971 \newcommand*{\glsentrysymbol}[1]{\csname glo@#1@symbol\endcsname}
```

`\Glsentrysymbol`

```
3972 \newrobustcmd*{\Glsentrysymbol}[1]{%
3973 \protected@edef\@glo@text{\csname glo@#1@symbol\endcsname}%
3974 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Plural form:

`\glsentrysymbolplural`

```
3975 \newcommand*{\glsentrysymbolplural}[1]{%
3976 \csname glo@#1@symbolplural\endcsname}
```

`\Glsentrysymbolplural`

```
3977 \newrobustcmd*{\Glsentrysymbolplural}[1]{%
3978 \protected@edef\@glo@text{\csname glo@#1@symbolplural\endcsname}%
3979 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Get the entry text to be used when the entry is first used in the document (as specified by the first key when the entry was defined).

`\glsentryfirst`

```
3980 \newcommand*{\glsentryfirst}[1]{\csname glo@#1@first\endcsname}
```

`\Glsentryfirst`

```
3981 \newrobustcmd*{\Glsentryfirst}[1]{%
3982 \protected@edef\@glo@text{\csname glo@#1@first\endcsname}%
3983 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

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

`\glsentryfirstplural`

```
3984 \newcommand*{\glsentryfirstplural}[1]{%
3985 \csname glo@#1@firstpl\endcsname}
```

`\Glsentryfirstplural`

```
3986 \newrobustcmd*{\Glsentryfirstplural}[1]{%
3987 \protected@edef\@glo@text{\csname glo@#1@firstpl\endcsname}%
3988 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

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

`\glentrytype`

3989 \newcommand*{\glentrytype}[1]{\csname glo@#1@type\endcsname}

Display the sort text used for this entry. Note that the sort key is sanitize, so unexpected results may occur if the sort key contained commands.

`\glentrysort`

3990 \newcommand*{\glentrysort}[1]{\csname glo@#1@sort\endcsname}

`\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.

3991 \newcommand*{\glentryuseri}[1]{\csname glo@#1@useri\endcsname}

`\Glsentryuseri`

3992 \newrobustcmd*{\Glsentryuseri}[1]{%
3993 \protected@edef\@glo@text{\csname glo@#1@useri\endcsname}%
3994 \expandafter\makefirstuc\expandafter{\@glo@text}}

`\glentryuserii` Get the second user key (as specified by the user2 when the entry was defined).
The argument is the label associated with the entry.

3995 \newcommand*{\glentryuserii}[1]{\csname glo@#1@userii\endcsname}

`\Glsentryuserii`

3996 \newrobustcmd*{\Glsentryuserii}[1]{%
3997 \protected@edef\@glo@text{\csname glo@#1@userii\endcsname}%
3998 \expandafter\makefirstuc\expandafter{\@glo@text}}

`\glentryuseriii` Get the third user key (as specified by the user3 when the entry was defined).
The argument is the label associated with the entry.

3999 \newcommand*{\glentryuseriii}[1]{\csname glo@#1@useriii\endcsname}

`\Glsentryuseriii`

4000 \newrobustcmd*{\Glsentryuseriii}[1]{%
4001 \protected@edef\@glo@text{\csname glo@#1@useriii\endcsname}%
4002 \expandafter\makefirstuc\expandafter{\@glo@text}}

`\glentryuseriv` Get the fourth user key (as specified by the user4 when the entry was defined).
The argument is the label associated with the entry.

4003 \newcommand*{\glentryuseriv}[1]{\csname glo@#1@useriv\endcsname}

`\Glsentryuseriv`

4004 \newrobustcmd*{\Glsentryuseriv}[1]{%
4005 \protected@edef\@glo@text{\csname glo@#1@useriv\endcsname}%
4006 \expandafter\makefirstuc\expandafter{\@glo@text}}

`\glentryuserv` Get the fifth user key (as specified by the user5 when the entry was defined).
The argument is the label associated with the entry.

4007 \newcommand*{\glentryuserv}[1]{\csname glo@#1@userv\endcsname}

```

\Glsentryuserv
4008 \newrobustcmd*{\Glsentryuserv}[1]{%
4009 \protected@edef\@glo@text{\csname glo@#1@userv\endcsname}%
4010 \expandafter\makefirstuc\expandafter{\@glo@text}}

\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.
4011 \newcommand*{\glsentryuservi}[1]{\csname glo@#1@uservi\endcsname}

\Glsentryuservi
4012 \newrobustcmd*{\Glsentryuservi}[1]{%
4013 \protected@edef\@glo@text{\csname glo@#1@uservi\endcsname}%
4014 \expandafter\makefirstuc\expandafter{\@glo@text}}

\glsentryshort  Get the short key (as specified by the short the entry was defined). The argu-
                  ment is the label associated with the entry.
4015 \newcommand*{\glsentryshort}[1]{\csname glo@#1@short\endcsname}

\Glsentryshort
4016 \newrobustcmd*{\Glsentryshort}[1]{%
4017 \protected@edef\@glo@text{\csname glo@#1@short\endcsname}%
4018 \expandafter\makefirstuc\expandafter{\@glo@text}}

\glsentryshortpl  Get the short plural key (as specified by the shortplural the entry was defined).
                   The argument is the label associated with the entry.
4019 \newcommand*{\glsentryshortpl}[1]{\csname glo@#1@shortpl\endcsname}

\Glsentryshortpl
4020 \newrobustcmd*{\Glsentryshortpl}[1]{%
4021 \protected@edef\@glo@text{\csname glo@#1@shortpl\endcsname}%
4022 \expandafter\makefirstuc\expandafter{\@glo@text}}

\glsentrylong  Get the long key (as specified by the long the entry was defined). The argument
                is the label associated with the entry.
4023 \newcommand*{\glsentrylong}[1]{\csname glo@#1@long\endcsname}

\Glsentrylong
4024 \newrobustcmd*{\Glsentrylong}[1]{%
4025 \protected@edef\@glo@text{\csname glo@#1@long\endcsname}%
4026 \expandafter\makefirstuc\expandafter{\@glo@text}}

\glsentrylongpl  Get the long plural key (as specified by the longplural the entry was defined).
                  The argument is the label associated with the entry.
4027 \newcommand*{\glsentrylongpl}[1]{\csname glo@#1@longpl\endcsname}

```

`\Glsentrylongpl`

```
4028 \newrobustcmd*{\Glsentrylongpl}[1]{%
4029 \protected@edef\@glo@text{\csname glo@#1@longpl\endcsname}%
4030 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Short cut macros to access full form:

`\glsentryfull`

```
4031 \newcommand*{\glsentryfull}[1]{%
4032 \acrfullformat{\glsentrylong{#1}}{\acronymfont{\glsentryshort{#1}}}%
4033 }
```

`\Glsentryfull`

```
4034 \newrobustcmd*{\Glsentryfull}[1]{%
4035 \acrfullformat{\Glsentrylong{#1}}{\acronymfont{\glsentryshort{#1}}}%
4036 }
```

`\glsentryfullpl`

```
4037 \newcommand*{\glsentryfullpl}[1]{%
4038 \acrfullformat{\glsentrylongpl{#1}}{\acronymfont{\glsentryshortpl{#1}}}%
4039 }
```

`\Glsentryfullpl`

```
4040 \newrobustcmd*{\Glsentryfullpl}[1]{%
4041 \acrfullformat{\Glsentrylongpl{#1}}{\acronymfont{\glsentryshortpl{#1}}}%
4042 }
```

`\glsentrynumberlist` Displays the number list as is.

```
4043 \newcommand*{\glsentrynumberlist}[1]{%
4044 \glsdoifexists{#1}%
4045 {%
4046 \csname glo@#1@numberlist\endcsname
4047 }%
4048 }
```

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

```
4049 \@ifpackageloaded{hyperref} {%
4050 \newcommand*{\glsdisplaynumberlist}[1]{%
4051 \GlossariesWarning
4052 {%
4053 \string\glsdisplaynumberlist\space
4054 doesn't work with hyperref.^^JUsing
4055 \string\glsentrynumberlist\space instead%
4056 }%
4057 \glsentrynumberlist{#1}%
4058 }%
4059 }%
4060 {%
```

```

4061 \newcommand*{\glsdisplaynumberlist}[1]{%
4062   \glsdoifexists{#1}%
4063   {%
4064     \bgroup
4065     \def\@glo@label{#1}%
4066     \let\@org@glsglnumberformat\glsglnumberformat
4067     \def\glsglnumberformat##1{##1}%
4068     \protected@edef\the@numberlist{\csname glo@\@glo@label @numberlist\endcsname}%
4069     \def\@glsgl@numlist@sep{}%
4070     \def\@glsgl@numlist@nextsep{}%
4071     \def\@glsgl@numlist@lastsep{}%
4072     \def\@glsgl@thislist{}%
4073     \def\@glsgl@donext@def{}%
4074     \renewcommand\do[1]{%
4075       \protected@edef\@glsgl@thislist{%
4076         \@glsgl@thislist
4077         \noexpand\@glsgl@numlist@sep
4078         ##1%
4079       }%
4080       \let\@glsgl@numlist@sep\@glsgl@numlist@nextsep
4081       \def\@glsgl@numlist@nextsep{\glsgl@numlist@sep}%
4082       \@glsgl@donext@def
4083       \def\@glsgl@donext@def{%
4084         \def\@glsgl@numlist@lastsep{\glsgl@numlist@lastsep}%
4085       }%
4086     }%
4087     \expandafter \glsgl@numlist@parser \expandafter{\the@numberlist}%
4088     \let\@glsgl@numlist@sep\@glsgl@numlist@lastsep
4089     \@glsgl@thislist
4090   \egroup
4091 }%
4092 }
4093 }

```

\glsgl@numlist@sep

```

4094 \newcommand*{\glsgl@numlist@sep}{, }

```

\glsgl@numlist@lastsep

```

4095 \newcommand*{\glsgl@numlist@lastsep}{ \& }

```

\glshyperlink Provide a hyperlink to a glossary entry without adding information to the glossary file. The entry needs to be added using a command like `\glslink` or `\glsadd` to ensure that the target is defined. The first (optional) argument specifies the link text. The entry name is used by default. The second argument is the entry label.

```

4096 \newcommand*{\glshyperlink}[2][\glsgl@entrytext{\@glo@label}]{%
4097 \def\@glo@label{#2}%
4098 \@glsgl@link{\glsgl@linkprefix#2}{#1}}

```

1.11 Adding an entry to the glossary without generating text

The following keys are provided for `\glsadd` and `\glsaddall`:

```
4099 \define@key{glossadd}{counter}{\def\@gls@counter{#1}}
```

```
4100 \define@key{glossadd}{format}{\def\@glsnumberformat{#1}}
```

This key is only used by `\glsaddall`:

```
4101 \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`

```
4102 \newrobustcmd*{\glsadd}[2][]{%
4103   \glsdoifexists{#2}%
4104   {%
4105     \def\@glsnumberformat{glsnumberformat}%
4106     \edef\@gls@counter{\csname glo@#2@counter\endcsname}%
4107     \setkeys{glossadd}{#1}%
```

Store the entry's counter in `\theglentrycounter`

```
4108     \@gls@saveentrycounter
4109     \@do@wrglossary{#2}%
4110   }%
4111 }
```

`\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`

```
4112 \newrobustcmd*{\glsaddall}[1][]{%
4113   \edef\@glo@type{\@glo@types}%
4114   \setkeys{glossadd}{#1}%
4115   \forallglsentries[\@glo@type]{\@glo@entry}{%
4116     \glsadd[#1]{\@glo@entry}%
4117   }%
4118 }
```

`\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.

```

4119 \newrobustcmd*{\glsaddallunused}[1][\@glo@types]{%
4120 \forallglsentries[#1]{\@glo@entry}%
4121 {%
4122 \ifglsused{\@glo@entry}{\glsadd[format=@gobble]{\@glo@entry}}%
4123 }%
4124 }

```

1.12 Creating associated files

The `\writeist` command creates the associated customized `.ist` `makeindex` style file. While defining this command, some characters have their catcodes temporarily changed to ensure they get written to the `.ist` file correctly. The `makeindex` actual character (usually `@`) is redefined to be a `?`, to allow internal commands to be written to the glossary file output file.

The special characters are stored in `\@gls@actualchar`, `\@gls@encapchar`, `\@gls@levelchar` and `\@gls@quotechar` to make them easier to use later, but don't change these values, because the characters are encoded in the command definitions that are used to escape the special characters (which means that the user no longer needs to worry about `makeindex` special characters).

The symbols and numbers label for group headings are hardwired into the `.ist` file as `glssymbols` and `glsnumbers`, the group titles can be translated (so that `\glssymbolsgroupname` replaces `glssymbols` and `\glsnumbersgroupname` replaces `glsnumbers`) using the command `\glsgetgrouptitle` which is defined in `.`. This is done to prevent any problem characters in `\glssymbolsgroupname` and `\glsnumbersgroupname` from breaking hyperlinks.

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

```

4125 \edef\glsopenbrace{\expandafter\@gobble\string\{ }

```

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

```

4126 \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.

```

4127 \edef\glsquote#1{\string"#1\string"}

```

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

```

4128 \ifglsxindy
4129 \newcommand*{\@glsfirstletter}{A}
4130 \fi

```

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

```

4131 \ifglsxindy

```

```

4132 \newcommand*{\GlsSetXdyFirstLetterAfterDigits}[1]{%
4133 \renewcommand*{\@glsfirstletter}{#1}}
4134 \else
4135 \newcommand*{\GlsSetXdyFirstLetterAfterDigits}[1]{%
4136 \glsnoxywarning\GlsSetXdyFirstLetterAfterDigits}
4137 \fi

```

`\@glsminrange` Define the minimum number of successive location references to merge into a range.

```

4138 \newcommand*{\@glsminrange}{2}

```

`\setXdyMinRangeLength` 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.

```

4139 \ifglxindy
4140 \newcommand*{\GlsSetXdyMinRangeLength}[1]{%
4141 \renewcommand*{\@glsminrange}{#1}}
4142 \else
4143 \newcommand*{\GlsSetXdyMinRangeLength}[1]{%
4144 \glsnoxywarning\GlsSetXdyMinRangeLength}
4145 \fi

```

`\writeist`

```

4146 \ifglxindy

```

Code to use if xindy is required.

```

4147 \def\writeist{%

```

Update attributes list

```

4148 \@gls@addpredefinedattributes

```

Open the file.

```

4149 \openout\glswrite=\istfilename

```

Write header comment at the start of the file

```

4150 \write\glswrite{;; xindy style file created by the glossaries
4151 package}%
4152 \write\glswrite{;; for document '\jobname' on
4153 \the\year-\the\month-\the\day}%

```

Specify the required styles

```

4154 \write\glswrite{^^J; required styles^^J}
4155 \@for\@xdystyle:=\@xdyrequiredstyles\do{%
4156 \ifx\@xdystyle\@empty
4157 \else
4158 \protected@write\glswrite{{(require
4159 \string"\@xdystyle.xdy\string")}}%
4160 \fi
4161 }%

```

List the allowed attributes (possible values used by the format key)

```
4162 \write\glswrite{^^J%
4163 ; list of allowed attributes (number formats)^^J}%
4164 \write\glswrite{(define-attributes ((\@xdyattributes)))}%
```

Define any additional alphabets

```
4165 \write\glswrite{^^J; user defined alphabets^^J}%
4166 \write\glswrite{\@xdyuseralphabets}%
```

Define location classes.

```
4167 \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.

```
4168 \@for\@gls@classI:=\@gls@xdy@locationlist\do{%
```

Case were $\langle Hprefix \rangle$ is empty:

```
4169 \protected@write\glswrite{}{(define-location-class
4170 \string"\@gls@classI\string"^^J\space\space\space
4171 (
4172 :sep "{{"
4173 \csname @gls@xdy@Lclass@\@gls@classI\endcsname\space
4174 :sep "}"
4175 )
4176 ^^J\space\space\space
4177 :min-range-length \@glsminrange^^J%
4178 )
4179 }%
```

Nested iteration over all classes:

```
4180 {%
4181 \@for\@gls@classII:=\@gls@xdy@locationlist\do{%
4182 \protected@write\glswrite{}{(define-location-class
4183 \string"\@gls@classII-\@gls@classI\string"
4184 ^^J\space\space\space
4185 (
4186 :sep "{"
4187 \csname @gls@xdy@Lclass@\@gls@classII\endcsname\space
4188 :sep "{{"
4189 \csname @gls@xdy@Lclass@\@gls@classI\endcsname\space
4190 :sep "}"
4191 )
4192 ^^J\space\space\space
4193 :min-range-length \@glsminrange^^J%
4194 )
4195 }%
4196 }%
4197 }%
4198 }%
```

User defined location classes (needs checking for new location format).

```

4199 \write\glswrite{^^J; user defined location classes}%
4200 \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.)

```

4201 \write\glswrite{^^J; define cross-reference class^^J}%
4202 \write\glswrite{(define-crossref-class \string"see\string"
4203 :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.)

```

4204 \write\glswrite{(markup-crossref-list
4205 :class \string"see\string"^^J\space\space\space
4206 :open \string"\string\glsseeformat\string"
4207 :close \string"{}\string")}%

```

List the order to sort the classes.

```

4208 \write\glswrite{^^J; define the order of the location classes}%
4209 \write\glswrite{(define-location-class-order
4210 (\@xdylocationclassorder))}%

```

Specify what to write to the start and end of the glossary file.

```

4211 \write\glswrite{^^J; define the glossary markup^^J}%

4212 \write\glswrite{(markup-index^^J\space\space\space
4213 :open \string"\string
4214 \glossarysection[\string\glossarytoctitle]{\string
4215 \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`)

```

4216 \@for\@this@ctr:=\@xdycounters\do{%
4217 {%
4218 \@for\@this@attr:=\@xdyattributelist\do{%
4219 \protected\write\glswrite{}{\string\providecommand*%
4220 \expandafter\string
4221 \csname glsX\@this@ctr X\@this@attr\endcsname[2]%
4222 {%
4223 \string\setentrycounter
4224 [\expandafter\@gobble\string\#1]{\@this@ctr}%
4225 \expandafter\string
4226 \csname\@this@attr\endcsname
4227 {\expandafter\@gobble\string\#2}%
4228 }%
4229 }%
4230 }%
4231 }%
4232 }%

```

Add the end part of the open tag and the rest of the markup-index information:

```
4233 \write\glswrite{%
4234     \string\begin
4235     {theglossary}\string\glossaryheader\string~n\string" ^^J\space
4236     \space\space:close \string"\expandafter\@gobble
4237     \string~n\string~n\string
4238     \end{theglossary}\string\glossarypostamble
4239     \string~n\string" ^^J\space\space\space
4240     :tree)}}%
```

Specify what to put between letter groups

```
4241 \write\glswrite{(markup-letter-group-list
4242     :sep \string"\string\glsgroupskip\string~n\string"))}%
```

Specify what to put between entries

```
4243 \write\glswrite{(markup-indexentry
4244     :open \string"\string\relax \string\glsresetentrylist
4245     \string~n\string"))}%
```

Specify how to format entries

```
4246 \write\glswrite{(markup-locclass-list :open
4247     \string"\glsopenbrace\string\glossaryentrynumbers
4248     \glsopenbrace\string\relax\space \string"^^J\space\space\space
4249     :sep \string", \string"
4250     :close \string"\glsclosebrace\glsclosebrace\string"))}%
```

Specify how to separate location numbers

```
4251 \write\glswrite{(markup-locref-list
4252     :sep \string"\string\delimN\space\string"))}%
```

Specify how to indicate location ranges

```
4253 \write\glswrite{(markup-range
4254     :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.

```
4255 \@onelevel@sanitize\gls@suffixF
4256 \@onelevel@sanitize\gls@suffixFF
4257 \ifx\gls@suffixF\@empty
4258 \else
4259     \write\glswrite{(markup-range
4260         :close "\gls@suffixF" :length 1 :ignore-end)}}%
4261 \fi
4262 \ifx\gls@suffixFF\@empty
4263 \else
4264     \write\glswrite{(markup-range
4265         :close "\gls@suffixFF" :length 2 :ignore-end)}}%
4266 \fi
```

Specify how to format locations.

```
4267 \write\glswrite{^^J; define format to use for locations^^J}%
4268 \write\glswrite{\@xdylocref}}%
```

Specify how to separate letter groups.

```
4269 \write\glswrite{^^J; define letter group list format^^J}%
4270 \write\glswrite{(markup-letter-group-list
4271 :sep \string"\string\glsgroupskip\string~n\string")}%
```

Define letter group headings.

```
4272 \write\glswrite{^^J; letter group headings^^J}%
4273 \write\glswrite{(markup-letter-group
4274 :open-head \string"\string\glsgroupheading
4275 \glsoopenbrace\string"^^J\space\space\space
4276 :close-head \string"\glsclosebrace\string")}%
```

Define additional letter groups.

```
4277 \write\glswrite{^^J; additional letter groups^^J}%
4278 \write\glswrite{\@xdylettergroups}%
```

Define additional sort rules

```
4279 \write\glswrite{^^J; additional sort rules^^J}
4280 \write\glswrite{\@xdysortrules}%
```

Close the style file

```
4281 \closeout\glswrite
```

Suppress any further calls.

```
4282 \let\writeist\relax
4283 }
4284 \else
```

Code to use if makeindex is required.

```
4285 \edef\@gls@actualchar{\string?}
4286 \edef\@gls@encapchar{\string|}
4287 \edef\@gls@levelchar{\string!}
4288 \edef\@gls@quotechar{\string"}
4289 \def\writeist{\relax
4290 \openout\glswrite=\istfilename
4291 \write\glswrite{\expandafter\@gobble\string}% makeindex style file
4292 created by the glossaries package}
4293 \write\glswrite{\expandafter\@gobble\string}% for document
4294 '\jobname' on \the\year-\the\month-\the\day}
4295 \write\glswrite{actual '\@gls@actualchar'}
4296 \write\glswrite{encap '\@gls@encapchar'}
4297 \write\glswrite{level '\@gls@levelchar'}
4298 \write\glswrite{quote '\@gls@quotechar'}
4299 \write\glswrite{keyword \string"\string\glossaryentry\string"}
4300 \write\glswrite{preamble \string"\string\glossarysection[\string
4301 \glossarytoctitle]{\string\glossarytitle}\string
4302 \glossarypreamble\string\n\string\begin{theglossary}\string
4303 \glossaryheader\string\n\string"}
4304 \write\glswrite{postamble \string"\string%\string\n\string
4305 \end{theglossary}\string\glossarypostamble\string\n
4306 \string"}
4307 \write\glswrite{group_skip \string"\string\glsgroupskip\string\n
```

```

4308     \string"}
4309 \write\glswrite{item_0 \string"\string%\string\n\string"}
4310 \write\glswrite{item_1 \string"\string%\string\n\string"}
4311 \write\glswrite{item_2 \string"\string%\string\n\string"}
4312 \write\glswrite{item_01 \string"\string%\string\n\string"}
4313 \write\glswrite{item_x1
4314     \string"\string\relax \string\glresetentrylist\string\n
4315     \string"}
4316 \write\glswrite{item_12 \string"\string%\string\n\string"}
4317 \write\glswrite{item_x2
4318     \string"\string\relax \string\glresetentrylist\string\n
4319     \string"}

4320 \write\glswrite{delim_0 \string"\string\{\string
4321     \glossaryentrynumbers\string\{\string\relax \string"}
4322 \write\glswrite{delim_1 \string"\string\{\string
4323     \glossaryentrynumbers\string\{\string\relax \string"}
4324 \write\glswrite{delim_2 \string"\string\{\string
4325     \glossaryentrynumbers\string\{\string\relax \string"}
4326 \write\glswrite{delim_t \string"\string\}\string\}\string"}
4327 \write\glswrite{delim_n \string"\string\delimN \string"}
4328 \write\glswrite{delim_r \string"\string\delimR \string"}
4329 \write\glswrite{headings_flag 1}
4330 \write\glswrite{heading_prefix
4331     \string"\string\glsgroupheading\string\{\string"}
4332 \write\glswrite{heading_suffix
4333     \string"\string\}\string\relax
4334     \string\glresetentrylist \string"}
4335 \write\glswrite{symhead_positive \string"glssymbols\string"}
4336 \write\glswrite{numhead_positive \string"glnumbers\string"}
4337 \write\glswrite{page_compositor \string"glscpositor\string"}
4338 \@gls@escbsdq\gls@suffixF
4339 \@gls@escbsdq\gls@suffixFF
4340 \ifx\gls@suffixF\@empty
4341 \else
4342     \write\glswrite{suffix_2p \string"\gls@suffixF\string"}
4343 \fi
4344 \ifx\gls@suffixFF\@empty
4345 \else
4346     \write\glswrite{suffix_3p \string"\gls@suffixFF\string"}
4347 \fi
4348 \closeout\glswrite
4349 \let\writeist\relax
4350 }
4351 \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`

```
4352 \newcommand{\noist}{%
```

Update attributes list

```
4353 \@gls@addpredefinedattributes
```

```
4354 \let\writeist\relax
```

```
4355 }
```

`\@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`

```
4356 \newcommand*{\@makeglossary}[1]{%
```

```
4357 \ifglossaryexists{#1}%
```

```
4358 {%
```

Only create a new write if `savewrites=false` otherwise create a token to collect the information.

```
4359 \ifglssavewrites
```

```
4360 \expandafter\newtoks\csname glo@#1@filetok\endcsname
```

```
4361 \else
```

```
4362 \expandafter\newwrite\csname glo@#1@file\endcsname
```

```
4363 \expandafter\@glsopenfile\csname glo@#1@file\endcsname{#1}%
```

```
4364 \fi
```

```
4365 \@gls@renewglossary
```

```
4366 \writeist
```

```
4367 }%
```

```
4368 {%
```

```
4369 \PackageError{glossaries}%
```

```
4370 {Glossary type ‘#1’ not defined}%
```

```
4371 {New glossaries must be defined before using \string\makeglossary}%
```

```
4372 }%
```

```
4373 }
```

`\@glsopenfile` Open write file associated with the given glossary.

```
4374 \newcommand*{\@glsopenfile}[2]{%
```

```
4375 \immediate\openout#1=\jobname.\csname @glotype@#2@out\endcsname
```

```
4376 \PackageInfo{glossaries}{Writing glossary file
```

```
4377 \jobname.\csname @glotype@#2@out\endcsname}%
```

```
4378 }
```

`\@nomakeglossaries` Issue warning that `\makeglossaries` hasn't been used.

```

4379 \newcommand*{\warn@nomakeglossaries}{%
4380   \GlossariesWarningNoLine{\string\makeglossaries\space
4381   hasn't been used,^^Jthe glossaries will not be updated}%
4382 }

```

\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

```

4383 \newcommand*{\makeglossaries}{%
  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.
4384   \protected@write\@auxout{}{\string\providecommand\string\@glsorder[1]{}%
4385   \protected@write\@auxout{}{\string\providecommand\string\@istfilename[1]{}%
4386 % Write the name of the style file to the aux file
4387 % (needed by \app{makeglossaries})
4388 %   \begin{macrocode}
4389   \protected@write\@auxout{}{\string\@istfilename{\istfilename}}%
4390   \protected@write\@auxout{}{\string\@glsorder{\glsorder}}%

```

Iterate through each glossary type and activate it.

```

4391   \@for\@glo@type:=\@glo@types\do{%
4392     \ifthenelse{\equal{\@glo@type}{}}{}{}%
4393     \@makeglossary{\@glo@type}%
4394   }%

```

New glossaries must be created before \makeglossaries so disable \newglossary.

```

4395   \renewcommand*\newglossary[4][]{%
4396   \PackageError{glossaries}{New glossaries
4397   must be created before \string\makeglossaries}{You need
4398   to move \string\makeglossaries\space after all your
4399   \string\newglossary\space commands}}%

```

Any subsequent instances of this command should have no effect

```

4400   \let\@makeglossary\relax
4401   \let\makeglossary\relax
4402   \let\makeglossaries\relax

```

Disable all commands that have no effect after \makeglossaries

```

4403   \@disable@onlypremakeg

```

Allow see key:

```

4404   \let\gls@checkseeallowed\relax

```

Suppress warning about no \makeglossaries

```

4405   \let\warn@nomakeglossaries\relax

```

Declare list parser for `\glsdisplaynumberlist`

```

4406 \ifglssavenumberlist
4407   \edef\@gls@doddeflistparser{\noexpand\DeclareListParser
4408     {\noexpand\glsnumlistparser}{\delimN}}}%
4409   \@gls@doddeflistparser
4410 \fi
4411 }
```

Must occur in the preamble:

```
4412 \@onlypreamble{\makeglossaries}
```

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`

```
4413 \let\makeglossary\makeglossaries
```

If `\makeglossaries` hasn't been used, issue a warning. Also issue a warning if neither `\printglossaries` nor `\printglossary` have been used.

```

4414 \AtEndDocument{%
4415   \warn@nomakeglossaries
4416   \warn@noprintglossary
4417 }
```

1.13 Writing information to associated files

`\glswrite` The write used for style file also used for all other output files if `savewrites=true`.

```
4418 \newwrite\glswrite
```

`\istfile` Deprecated.

```
4419 \def\istfile{\glswrite}
```

At the end of the document, the files should be created if `savewrites=true`.

```

4420 \AtEndDocument{%
4421   \glswritefiles
4422 }
```

`\@glswritefiles` Only write the files if `savewrites=true`

```
4423 \newcommand*{\@glswritefiles}{%
```

Iterate through all the glossaries

```
4424 \forallglossaries{\@glo@type}{%
```

Check for empty glossaries (patch provided by Patrick Häcker)

```

4425   \ifcsundef{glo@\@glo@type @filetok}%
4426   {%
4427     \def\gls@tmp{}%
4428   }%
```

```

4429     {%
4430         \edef\gls@tmp{\expandafter\the
4431             \csname glo@\@glo@type @filetok\endcsname}%
4432     }%
4433     \ifx\gls@tmp\@empty
4434         \ifx\@glo@type\glsdefaulttype
4435             \GlossariesWarningNoLine{Glossary '@@glo@type' has no
4436                 entries.^^JRemember to use package option 'nomain' if
4437 you
4438                 don't want to^^Juse the main glossary}%
4439         \else
4440             \GlossariesWarningNoLine{Glossary '@@glo@type' has no
4441                 entries}%
4442         \fi
4443     \else
4444         \@glsopenfile{\glswrite}{\@glo@type}%
4445         \immediate\write\glswrite{%
4446             \expandafter\the
4447             \csname glo@\@glo@type @filetok\endcsname}%
4448         \immediate\closeout\glswrite
4449     \fi
4450 }%
4451 }

```

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`

```

4452 \renewcommand*{\glossary}[1][\glsdefaulttype]{%
4453     \@glossary[#1]%
4454 }

```

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`

```

4455 \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`

```

4456 \newcommand{\@gls@renewglossary}{%
4457     \gdef\@glossary[##1]{\@bsphack\begingroup\@wrglossary{##1}}%
4458     \let\@gls@renewglossary\@empty
4459 }

```

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`

```

4460 \renewcommand*{\@wrglossary}[2]{%
4461   \ifglssavewrites
4462     \protected@edef\@gls@tmp{\the\csname glo@#1@filetok\endcsname#2}%
4463     \expandafter\global\expandafter\csname glo@#1@filetok\endcsname
4464       \expandafter{\@gls@tmp^^J}%
4465   \else
4466     \ifcsdef{glo@#1@file}%
4467       {%
4468         \expandafter\protected@write\csname glo@#1@file\endcsname{%
4469           \gls@disablepagerefexpansion}{#2}%
4470       }%
4471       {%
4472         \GlossariesWarning{No file defined for glossary ‘#1’}%
4473       }%
4474   \fi
4475   \endgroup\@esphack
4476 }
```

`\@do@wrglossary`

```

4477 \newcommand*{\@do@wrglossary}[1]{%
4478   \ifglsindexonlyfirst
4479     \ifglsused{#1}{\@do@wrglossary{#1}}%
4480   \else
4481     \@do@wrglossary{#1}%
4482   \fi
4483 }
```

`@protected@pagefmts` List of page formats to be protected against expansion.

```

4484 \newcommand{\gls@protected@pagefmts}{%
4485   \gls@numberpage,\gls@alphpage,\gls@Alphpage,\gls@romanpage,\gls@Romanpage%
4486 }
```

`blepagerefexpansion`

```

4487 \newcommand*{\gls@disablepagerefexpansion}{%
4488   \@for\@gls@this:=\gls@protected@pagefmts\do
4489   {%
4490     \expandafter\let\@gls@this\relax
4491   }%
4492 }
```

`\gls@alphpage`

```

4493 \newcommand*{\gls@alphpage}{\@alph\c@page}
```

```

\gls@Alphpage
4494 \newcommand*{\gls@Alphpage}{\@Alph\c@page}

\gls@numberpage
4495 \newcommand*{\gls@numberpage}{\number\c@page}

\gls@romanpage
4496 \newcommand*{\gls@romanpage}{\romannumeral\c@page}

\gls@Romanpage
4497 \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.
4498 \newcommand*{\@do@wrglossary}[1]{%
4499 \begingroup
First a bit of hackery to prevent premature expansion of \c@page. Store original
definitions:
4500 \let\orgthe\the
4501 \let\orgnumber\number
4502 \let\orgromannumeral\romannumeral
4503 \let\orgalph\@alph
4504 \let\orgAlph\@Alph
4505 \let\orgRoman\@Roman
Redefine:
4506 \def\the##1{%
4507 \ifx##1\c@page \gls@numberpage\else\orgthe##1\fi}%
4508 \def\number##1{%
4509 \ifx##1\c@page \gls@numberpage\else\orgnumber##1\fi}%
4510 \def\romannumeral##1{%
4511 \ifx##1\c@page \gls@romanpage\else\orgromannumeral##1\fi}%
4512 \def\@Roman##1{%
4513 \ifx##1\c@page \gls@Romanpage\else\orgRoman##1\fi}%
4514 \def\@alph##1{%
4515 \ifx##1\c@page \gls@alphpage\else\orgalph##1\fi}%
4516 \def\@Alph##1{%
4517 \ifx##1\c@page \gls@Alphpage\else\orgAlph##1\fi}%
Prevent expansion:
4518 \gls@disablepagerefexpansion
Now store location in \glslocref:
4519 \protected@xdef\@glslocref{\theglsentrycounter}%
4520 \endgroup
Escape any special characters
4521 \gls@checkmkidxchars\@glslocref

```

Check if the hyper-location is the same as the location and set the hyper prefix.

```

4522 \expandafter\ifx\theHglentrycounter\theglsentrycounter
4523 \def\@glo@counterprefix{%
4524 \else
4525 \protected@edef\@glsHlocref{\theHglentrycounter}%
4526 \@gls@checkmkidxchars\@glsHlocref
4527 \edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix
4528 {\@glslocref}{\@glsHlocref}%
4529 }%
4530 \@do@gls@getcounterprefix
4531 \fi

```

Determine whether to use xindy or makeindex syntax

```

4532 \ifglxindy

```

Need to determine if the formatting information starts with a (or) indicating a range.

```

4533 \expandafter\@glo@check@mkidxrangechar\@glsnumberformat\@nil
4534 \def\@glo@range{%
4535 \expandafter\if\@glo@prefix(\relax
4536 \def\@glo@range{:open-range}%
4537 \else
4538 \expandafter\if\@glo@prefix)\relax
4539 \def\@glo@range{:close-range}%
4540 \fi
4541 \fi

```

Write to the glossary file using xindy syntax.

```

4542 \glossary[\csname glo@#1@type\endcsname]{%
4543 (indexentry :tkey (\csname glo@#1@index\endcsname)
4544 :locref \string{\@glo@counterprefix}{\@glslocref}\string" %
4545 :attr \string"\@gls@counter\@glo@suffix\string"
4546 \@glo@range
4547 )
4548 }%
4549 \else

```

Convert the format information into the format required for makeindex

```

4550 \@set@glo@numformat{\@glo@numfmt}{\@gls@counter}{\@glsnumberformat}%
4551 {\@glo@counterprefix}%

```

Write to the glossary file using makeindex syntax.

```

4552 \glossary[\csname glo@#1@type\endcsname]{%
4553 \string\glossaryentry{\csname glo@#1@index\endcsname
4554 \@gls@encapchar\@glo@numfmt}{\@glslocref}}%
4555 \fi
4556 }

```

`\@gls@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.)

```

4557 \newcommand*\@gls@getcounterprefix[2]{%
4558   \edef\@gls@thisloc{#1}\edef\@gls@thisHloc{#2}%
4559   \ifx\@gls@thisloc\@gls@thisHloc
4560     \def\@glo@counterprefix{}%
4561   \else
4562     \def\@gls@get@counterprefix##1.#1##2\end@getprefix{%
4563       \def\@glo@tmp{##2}%
4564       \ifx\@glo@tmp\@empty
4565         \def\@glo@counterprefix{}%
4566       \else
4567         \def\@glo@counterprefix{##1}%
4568       \fi
4569     }%
4570     \@gls@get@counterprefix#2.#1\end@getprefix
4571   \fi
4572 }
```

1.14 Glossary Entry Cross-References

`\@do@seeglossary` Write the glossary entry with a cross reference. The first argument is the entry's label, the second must be in the form `[\<tag>]{\<list>}`, where `\<tag>` is a tag such as “see” and `\<list>` is a list of labels.

```

4573 \newcommand{\@do@seeglossary}[2]{%
4574   \def\@gls@xref{#2}%
4575   \@onelevel@sanitize\@gls@xref
4576   \@gls@checkmkidxchars\@gls@xref
4577   \ifglxindy
4578     \glossary[\csname glo@#1@type\endcsname]{%
4579       (indexentry
4580         :tkey (\csname glo@#1@index\endcsname)
4581         :xref (\string"\@gls@xref\string")
4582         :attr \string"see\string"
4583       )
4584     }%
4585   \else
4586     \glossary[\csname glo@#1@type\endcsname]{%
4587       \string\glossaryentry{\csname glo@#1@index\endcsname
4588         \@gls@encapchar glsseeformat\@gls@xref}{Z}}%
4589     \fi
4590 }
```

`\@gls@fixbraces` If no optional argument is specified, list needs to be enclosed in a set of braces.

```

4591 \def\@gls@fixbraces#1#2#3\@nil{%
```

```

4592 \ifx#2[\relax
4593 \def#1{#2#3}%
4594 \else
4595 \def#1{{#2#3}}%
4596 \fi
4597 }

```

`\glssee` `\glssee{<label>}{<cross-ref list>}`

```

4598 \DeclareRobustCommand*\glssee[3][\seename]{%
4599 \do@seeglossary{#2}{#1}{#3}}
4600 \newcommand*\@glssee[3][\seename]{%
4601 \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.

```

4602 \DeclareRobustCommand*\glsseeformat[3][\seename]{%
4603 \emph{#1} \glsseelist{#2}}

```

`\glsseelist` `\glsseelist{<list>}` formats list of entry labels.

```

4604 \DeclareRobustCommand*\glsseelist[1]{%

```

If there is only one item in the list, set the last separator to do nothing.

```

4605 \let\@gls@dolast\relax

```

Don’t display separator on the first iteration of the loop

```

4606 \let\@gls@donext\relax

```

Iterate through the labels

```

4607 \@for\@gls@thislabel:=#1\do{%

```

Check if on last iteration of loop

```

4608 \ifx\@xfor@nextelement\@nnil

```

```

4609 \@gls@dolast

```

```

4610 \else

```

```

4611 \@gls@donext

```

```

4612 \fi

```

display the entry for this label

```

4613 \glsseeitem{\@gls@thislabel}%

```

Update separators

```

4614 \let\@gls@dolast\glsseelastsep

```

```

4615 \let\@gls@donext\glsseesep

```

```

4616 }%

```

```

4617 }

```

`\glsseelastsep` Separator to use between penultimate and ultimate entries in a cross-referencing list.

```

4618 \newcommand*\glsseelastsep{\space\andname\space}

```

`\glsseesep` Separator to use between entires in a cross-referencing list.

```
4619 \newcommand*\glsseesep}{, }
```

`\glsseeitem` `\glsseeitem{<label>}` formats individual entry in a cross-referencing list.

```
4620 \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.)

```
4621 \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.

```
4622 \newcommand*\gls@save@numberlist[1]{%
4623   \ifglssavenumberlist
4624     \toks@{#1}%
4625     \edef\@do@writeaux@info{%
4626       \noexpand\csgdef{glo@\glscurrententrylabel @numberlist}{\the\toks@}%
4627     }%
4628     \@onelevel@sanitize\@do@writeaux@info
4629     \protected@write\@auxout{}\@do@writeaux@info}%
4630   \fi
4631 }
```

`\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.)

```
4632 \def\warn@noprintglossary{%
4633   \GlossariesWarningNoLine{No \string\printglossary\space
4634     or \string\printglossaries\space
4635     found.^^JThis document will not have a glossary}%
4636 }
```

`\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.

```
4637 \ifcsundef{printglossary}{}%
4638 {%
```

If `\printglossary` is already defined, issue a warning and undefine it.

```
4639   \GlossariesWarning{Overriding \string\printglossary}%
```

```

4640 \undef\printglossary
4641 }

\printglossary has an optional argument. The default value is to set the glossary type to the main glossary.
4642 \newcommand*{\printglossary}[1][type=\glsdefaulttype]{%
    Set up defaults.
4643 \def\@glo@type{\glsdefaulttype}%
4644 \def\glossarytitle{\csname @glo@type\@glo@type @title\endcsname}%

4645 \def\glossarytoctitle{\glossarytitle}%
4646 \let\org@glossarytitle\glossarytitle
4647 \def\@glossarystyle{}%
4648 \def\gls@dotoc@title{\glssettoc@title{\@glo@type}}%

    Store current value of \glossaryentrynumbers. (This may be changed via the optional argument)
4649 \let\@org@glossaryentrynumbers\glossaryentrynumbers

    Localise the effects of the optional argument
4650 \bgroup

    Determine settings specified in the optional argument.
4651 \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)
4652 \ifx\glossarytitle\org@glossarytitle
4653 \else
4654 \expandafter\let\csname @glo@type\@glo@type @title\endcsname
4655 \glossarytitle
4656 \fi

    Allow a high-level user command to indicate the current glossary
4657 \let\currentglossary\@glo@type

    Enable individual number lists to be suppressed.
4658 \let\org@glossaryentrynumbers\glossaryentrynumbers
4659 \let\glsnonextpages\@glsnonextpages

    Enable individual number list to be activated:
4660 \let\glsnextpages\@glsnextpages

    Enable suppression of description terminators.
4661 \let\nopostdesc\@nopostdesc

    Set up the entry for the TOC
4662 \gls@dotoc@title

    Set the glossary style
4663 \@glossarystyle

```

added a way to fetch the current entry label (v3.08 updated for new `\glossentry` and `\subglossentry`):

```

4664 \let\gls@org@glossaryentryfield\glossentry
4665 \let\gls@org@glossarysubentryfield\subglossentry
4666 \renewcommand{\glossentry}[1]{%
4667   \gdef\glscurrententrylabel{##1}%
4668   \gls@org@glossaryentryfield{##1}%
4669 }%
4670 \renewcommand{\subglossentry}[2]{%
4671   \gdef\glscurrententrylabel{##2}%
4672   \gls@org@glossarysubentryfield{##1}{##2}%
4673 }%

```

Some macros may end up being expanded into internals in the glossary, so need to make @ a letter.

```

4674 \makeatletter

```

Input the glossary file, if it exists.

```

4675 \input@{\jobname.\csname @glo@type@\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.

```

4676 \IfFileExists{\jobname.\csname @glo@type@\glo@type @in\endcsname}%
4677 {}%
4678 {\null}%

```

If xindy is being used, need to write the language dependent information to the .aux file for makeglossaries.

```

4679 \ifglsxindy
4680 \ifcsundef{@xdy@\glo@type @language}%
4681 {%
4682   \edef\@do@auxoutstuff{%
4683     \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.

```

4684       \noexpand\immediate\noexpand\write\@auxout{%
4685         \string\providecommand\string\@xdylanguage[2]{}}%
4686       \noexpand\immediate\noexpand\write\@auxout{%
4687         \string\@xdylanguage{\glo@type}{\@xdy@main@language}}%
4688     }%
4689   }%
4690 }%
4691 {%
4692   \edef\@do@auxoutstuff{%
4693     \noexpand\AtEndDocument{%
4694       \noexpand\immediate\noexpand\write\@auxout{%
4695         \string\providecommand\string\@xdylanguage[2]{}}%

```

```

4696          \noexpand\immediate\noexpand\write\@auxout{%
4697          \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type
4698          @language\endcsname}}}%
4699      }%
4700  }%
4701  }%
4702  \@do@auxoutstuff
4703  \edef\@do@auxoutstuff{%
4704  \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.

```

4705          \noexpand\immediate\noexpand\write\@auxout{%
4706          \string\providecommand\string\@gls@codepage[2]{}}%
4707          \noexpand\immediate\noexpand\write\@auxout{%
4708          \string\@gls@codepage{\@glo@type}{\gls@codepage}}}%
4709      }%
4710  }%
4711  \@do@auxoutstuff
4712  \fi
4713  \egroup

```

Reset \glossaryentrynumbers

```

4714  \global\let\glossaryentrynumbers\org@glossaryentrynumbers

```

Suppress warning about no \printglossary

```

4715  \global\let\warn@noprintglossary\relax
4716 }

```

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

```

4717 \newcommand*\printglossaries{%
4718   \foralllglossaries{\@glo@type}{\printglossary[type=\@glo@type]}%
4719 }

```

The keys that can be used in the optional argument to \printglossary are as follows: The type key sets the glossary type.

```

4720 \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.

```

4721 \define@key{printgloss}{title}{%
4722   \def\glossarytitle{#1}%
4723   \let\gls@dotocitle\relax
4724 }

```

The toctitle sets the text used for the relevant entry in the table of contents.

```

4725 \define@key{printgloss}{toctitle}{%
4726   \def\glossarytoctitle{#1}%
4727   \let\gls@dotocitle\relax
4728 }

```

The style key sets the glossary style (but only for the given glossary).

```

4729 \define@key{printgloss}{style}{%
4730   \ifcsundef{@glsstyle@#1}%
4731   {%
4732     \PackageError{glossaries}%
4733       {Glossary style ‘#1’ undefined}{}%
4734   }%
4735   {%
4736     \def\@glossarystyle{\setglossentrycompatibility
4737       \csname @glsstyle@#1\endcsname}%
4738   }%
4739 }

```

The numberedsection key determines if this glossary should be in a numbered section.

```

4740 \define@choicekey{printgloss}{numberedsection}[\val\nr]{%
4741   false,nolabel,autolabel,nameref}[nolabel]{%
4742   \ifcase\nr\relax
4743     \renewcommand*{\@glossarysecstar}{*}%
4744     \renewcommand*{\@glossaryseclabel}{}%
4745   \or
4746     \renewcommand*{\@glossarysecstar}{}%
4747     \renewcommand*{\@glossaryseclabel}{}%
4748   \or
4749     \renewcommand*{\@glossarysecstar}{}%
4750     \renewcommand*{\@glossaryseclabel}{\label{\glsautoprefix\@glo@type}}%
4751   \or
4752     \renewcommand*{\@glossarysecstar}{*}%
4753     \renewcommand*{\@glossaryseclabel}{%
4754       \protected@edef\@currentlabelname{\glossarytoctitle}%
4755       \label{\glsautoprefix\@glo@type}}%
4756   \fi
4757 }

```

The nogroupskip key determines whether or not there should be a vertical gap between glossary groups.

```

4758 \define@choicekey{printgloss}{nogroupskip}{true,false}[true]{%
4759   \csuse{glsnogroupskip#1}%
4760 }

```

The nonumberlist key determines if this glossary should have a number list.

```

4761 \define@boolkey{printgloss}[gls]{nonumberlist}[true]{%
4762 \ifglsnonumberlist
4763   \def\glossaryentrynumbers##1{}%
4764 \else
4765   \def\glossaryentrynumbers##1{##1}%
4766 \fi}

```

`\@glsnonextpages` Suppresses the next number list only. Global assignments required as it may not occur in the same level of grouping as the next numberlist. (For example, if `\glsnonextpages` is place in the entry's description and 3 column tabular style glossary is used.) `\org@glossaryentrynumbers` needs to be set at the start of each glossary, in the event that `\glossaryentrynumber` is redefined.

```

4767 \newcommand*{\@glsnonextpages}{%
4768   \gdef\glossaryentrynumbers##1{%
4769     \glsresetentrylist
4770   }%
4771 }

```

`\@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.

```

4772 \newcommand*{\@glsnextpages}{%
4773   \gdef\glossaryentrynumbers##1{%
4774     ##1\glsresetentrylist}}

```

`\glsresetentrylist` Resets `\glossaryentrynumbers`

```

4775 \newcommand*{\glsresetentrylist}{%
4776   \global\let\glossaryentrynumbers\org@glossaryentrynumbers}

```

`\glsnonextpages` Outside of `\printglossary` this does nothing.

```

4777 \newcommand*{\glsnonextpages}{}

```

`\glsnextpages` Outside of `\printglossary` this does nothing.

```

4778 \newcommand*{\glsnextpages}{}

```

`glossaryentry` If the entrycounter package option has been used, define a counter to number each level 0 entry.

```

4779 \ifglsentrycounter
4780   \ifx\@gls@counterwithin\@empty
4781     \newcounter{glossaryentry}
4782   \else
4783     \newcounter{glossaryentry}[\@gls@counterwithin]
4784   \fi
4785   \def\theHglossaryentry{\currentglossary.\theglossaryentry}
4786 \fi

```

glossarysubentry If the subentrycounter package option has been used, define a counter to number each level 1 entry.

```

4787 \ifglssubentrycounter
4788   \ifglsentrycounter
4789     \newcounter{glossarysubentry}[glossaryentry]
4790   \else
4791     \newcounter{glossarysubentry}
4792   \fi
4793   \def\theHglossarysubentry{\currentglssubentry.\theglossarysubentry}
4794 \fi

```

resetsubentrycounter Resets the glossarysubentry counter.

```

4795 \ifglssubentrycounter
4796   \newcommand*{\glsresetsubentrycounter}{%
4797     \setcounter{glossarysubentry}{0}%
4798   }
4799 \else
4800   \newcommand*{\glsresetsubentrycounter}{}
4801 \fi

```

resetsubentrycounter Resets the glossareentry counter.

```

4802 \ifglsentrycounter
4803   \newcommand*{\glsresetentrycounter}{%
4804     \setcounter{glossaryentry}{0}%
4805   }
4806 \else
4807   \newcommand*{\glsresetentrycounter}{}
4808 \fi

```

\glsstepentry Advance the glossaryentry counter if in use. The argument is the label associated with the entry.

```

4809 \ifglsentrycounter
4810   \newcommand*{\glsstepentry}[1]{%
4811     \refstepcounter{glossaryentry}%
4812     \label{glsentry-#1}%
4813   }
4814 \else
4815   \newcommand*{\glsstepentry}[1]{}
4816 \fi

```

\glsstepsubentry Advance the glossarysubentry counter if in use. The argument is the label associated with the subentry.

```

4817 \ifglssubentrycounter
4818   \newcommand*{\glsstepsubentry}[1]{%
4819     \def\currentglssubentry{#1}%
4820     \refstepcounter{glossarysubentry}%
4821     \label{glsentry-#1}%
4822   }

```

```

4823 \else
4824   \newcommand*{\glsstepsubentry}[1]{%
4825 \fi

```

`\glsrefentry` Reference the entry or sub-entry counter if in use, otherwise just do `\gls`.

```

4826 \ifglsentrycounter
4827   \newcommand*{\glsrefentry}[1]{\ref{glsentry-#1}}
4828 \else
4829   \ifglssubentrycounter
4830     \newcommand*{\glsrefentry}[1]{\ref{glsentry-#1}}
4831   \else
4832     \newcommand*{\glsrefentry}[1]{\gls{#1}}
4833   \fi
4834 \fi

```

`glsentrycounterlabel` Defines how to display the glossaryentry counter.

```

4835 \ifglsentrycounter
4836   \newcommand*{\glsentrycounterlabel}{\theglossaryentry.\space}
4837 \else
4838   \newcommand*{\glsentrycounterlabel}{}
4839 \fi

```

`glssubentrycounterlabel` Defines how to display the glossarysubentry counter.

```

4840 \ifglssubentrycounter
4841   \newcommand*{\glssubentrycounterlabel}{\theglossarysubentry)\space}
4842 \else
4843   \newcommand*{\glssubentrycounterlabel}{}
4844 \fi

```

`\glsentryitem` Step and display glossaryentry counter, if appropriate.

```

4845 \ifglsentrycounter
4846   \newcommand*{\glsentryitem}[1]{%
4847     \glsstepentry{#1}\glsentrycounterlabel
4848   }
4849 \else
4850   \newcommand*{\glsentryitem}[1]{\glsresetsubentrycounter}
4851 \fi

```

`\glssubentryitem` Step and display glossarysubentry counter, if appropriate.

```

4852 \ifglssubentrycounter
4853   \newcommand*{\glssubentryitem}[1]{%
4854     \glsstepsubentry{#1}\glssubentrycounterlabel
4855   }
4856 \else
4857   \newcommand*{\glssubentryitem}[1]{}
4858 \fi

```

`theglossary` If the `theglossary` environment has already been defined, a warning will be issued. This environment should be redefined by glossary styles.

```

4859 \ifcsundef{theglossary}%
4860 {%
4861   \newenvironment{theglossary}{}{}%
4862 }%
4863 {%
4864   \GlossariesWarning{overriding ‘theglossary’ environment}%
4865   \renewenvironment{theglossary}{}{}%
4866 }

```

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`

```

4867 \newcommand*{\glossaryheader}{}

```

`\glstarget` `\glstarget{<label>}{<name>}`

Provide user interface to `\@glstarget` to make it easier to modify the glossary style in the document.

```

4868 \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>}`

```

4869 \providecommand*{\compatibleglossentry}[2]{%
4870   \toks@{#2}%
4871   \protected@edef\@do@glossentry{\noexpand\glossaryentryfield{#1}%
4872     {\noexpand\glsnamefont
4873       {\expandafter\expandonce\csname glo@#1@name\endcsname}}%
4874     {\expandafter\expandonce\csname glo@#1@desc\endcsname}%
4875     {\expandafter\expandonce\csname glo@#1@symbol\endcsname}%
4876     {\the\toks@}}%
4877   }%
4878   \@do@glossentry
4879 }

```

`\glossentryname`

```

4880 \newcommand*{\glossentryname}[1]{%
4881   \glsdoifexists{#1}%

```

```

4882  {%
4883    \letcs{\glo@name}{glo@#1@name}%
4884    \expandafter\glsnamefont\expandafter{\glo@name}%
4885  }%
4886 }

```

\Glossentryname

```

4887 \newcommand*{\Glossentryname}[1]{%
4888   \glsdoifexists{#1}%
4889   {%
4890     \glsnamefont{\Glsentryname{#1}}%
4891   }%
4892 }

```

\glossentrydesc

```

4893 \newcommand*{\glossentrydesc}[1]{%
4894   \glsdoifexists{#1}%
4895   {%
4896     \glentrydesc{#1}%
4897   }%
4898 }

```

\Glossentrydesc

```

4899 \newcommand*{\Glossentrydesc}[1]{%
4900   \glsdoifexists{#1}%
4901   {%
4902     \Glsentrydesc{#1}%
4903   }%
4904 }

```

\glossentrysymbol

```

4905 \newcommand*{\glossentrysymbol}[1]{%
4906   \glsdoifexists{#1}%
4907   {%
4908     \glentrysymbol{#1}%
4909   }%
4910 }

```

\Glossentrysymbol

```

4911 \newcommand*{\Glossentrysymbol}[1]{%
4912   \glsdoifexists{#1}%
4913   {%
4914     \Glsentrysymbol{#1}%
4915   }%
4916 }

```

patiblesubglossentry `\subglossentry{<level>}{<label>}{<page-list>}`

```

4917 \providecommand*\compatiblesubglossentry}[3]{%
4918   \toks@{#3}%
4919   \protected@edef\@do@subglossentry{\noexpand\glossarysubentryfield{\number#1}%
4920     {#2}%
4921     {\noexpand\glsnamefont
4922       {\expandafter\expandonce\csname glo@#2@name\endcsname}}%
4923     {\expandafter\expandonce\csname glo@#2@desc\endcsname}%
4924     {\expandafter\expandonce\csname glo@#2@symbol\endcsname}%
4925     {\the\toks@}%
4926   }%
4927   \@do@subglossentry
4928 }

```

sentrycompatibility

```

4929 \newcommand*\setglossentrycompatibility{%
4930   \let\glossentry\compatibleglossentry
4931   \let\subglossentry\compatiblesubglossentry
4932 }
4933 \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.

```

4934 \newcommand{\glossaryentryfield}[5]{%
4935   \GlossariesWarning
4936   {Deprecated use of \string\glossaryentryfield.^^J
4937     I recommend you change to \string\glossentry.^^J
4938     If you've just upgraded, try removing your gls auxiliary
4939     files^^J and recompile}%
4940   \noindent\textbf{\glstarget{#1}{#2}} #4 #3. #5\par}

```

glossarysubentryfield \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 <symbol>. The first argument is a number indicating the level. (The level should be greater than or equal to 1.)

```

4941 \newcommand*\glossarysubentryfield}[6]{%
4942   \GlossariesWarning
4943   {Deprecated use of \string\glossarysubentryfield.^^J
4944     I recommend you change to \string\subglossentry.^^J
4945     If you've just upgraded, try removing your gls auxiliary
4946     files^^J and recompile}%
4947   \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`

```
4948 \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`

```
4949 \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 `<label>`. 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: `glssymbols`, `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`

```
4950 \newcommand*{\glsgetgrouptitle}[1]{%
4951   \@gls@getgrouptitle{#1}{\@gls@grptitle}%
4952   \@gls@grptitle
4953 }
```

`\@gls@getgrouptitle` Gets the group title specified by the label (first argument) and stores in the second argument, which must be a control sequence.

```
4954 \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.

```
4955 \dtl@ifsingle{#1}%
4956 {%
4957   \ifcsundef{#1groupname}{\def#2{#1}}{\letcs#2{#1groupname}}%
4958 }%
4959 {%
4960   \ifboolexpr{test{\ifstrequal{#1}{glssymbols}}
4961               or test{\ifstrequal{#1}{glsnumbers}}}%
4962   {%
4963     \ifcsundef{#1groupname}{\def#2{#1}}{\letcs#2{#1groupname}}%
4964   }%
4965   {%
4966     \def#2{#1}%
4967   }%
4968 }%
4969 }
```

`\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`

```
4970 \newcommand*{\glsgetgrouplabel}[1]{%
4971 \ifthenelse{\equal{#1}{\glssymbolsgroupname}}{\glssymbols}{%
4972 \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`

```
4973 \newcommand*{\setentrycounter}[2][ ]{%
4974   \def\@glo@counterprefix{#1}%
4975   \ifx\@glo@counterprefix\@empty
4976     \def\@glo@counterprefix{.}%
4977   \else
4978     \def\@glo@counterprefix{.#1.}%
4979   \fi
4980   \def\glsentrycounter{#2}%
4981 }
```

The current glossary style can be set using `\setglossarystyle{<style>}`.

`\setglossarystyle`

```
4982 \newcommand*{\setglossarystyle}[1]{%
4983   \ifcsundef{@glsstyle@#1}%
4984   {%
4985     \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
4986   }%
4987   {%
4988     \csname @glsstyle@#1\endcsname
4989   }%
4990 }
```

`\glossarystyle`

```
4991 \newcommand*{\glossarystyle}[1]{%
4992   \ifcsundef{@glsstyle@#1}%
4993   {%
4994     \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
4995   }%
4996   {%
4997     \GlossariesWarning
4998     {Deprecated command \string\glossarystyle.^^J
4999     I recommend you switch to \string\setglossarystyle\space unless
5000     you want to maintain backward compatibility}%
5001     \setglossentrycompatibility
5002     \csname @glsstyle@#1\endcsname

5003     \ifcsdef{@glscompstyle@#1}%
5004     {\setglossentrycompatibility\csuse{@glscompstyle@#1}}%
5005     {}%
5006   }%
5007 }
```

`\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.

```
5008 \newcommand{\newglossarystyle}[2]{%
5009   \ifcsundef{@glsstyle@#1}%
5010   {%
5011     \expandafter\def\csname @glsstyle@#1\endcsname{#2}%
5012   }%
5013   {%
5014     \PackageError{glossaries}{Glossary style ‘#1’ is already defined}{}%
5015   }%
5016 }
```

`\renewglossarystyle` Code for this macro supplied by Marco Daniel.

```
5017 \newcommand{\renewglossarystyle}[2]{%
5018   \ifcsundef{@glsstyle@#1}%
5019   {%
5020     \PackageError{glossaries}{Glossary style ‘#1’ isn’t already defined}{}%
5021   }%
5022   {%
5023     \csdef{@glsstyle@#1}{#2}%
5024   }%
5025 }
```

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`

```
5026 \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`

```
5027 \ifcsundef{hyperlink}%
5028 {%
5029   \def\glshypernumber#1{#1}%
5030 }%
5031 {%
5032   \def\glshypernumber#1{\@glshypernumber#1\nohyperpage{}}\@nil}
5033 }
```

`\@glshypernumber` This code was provided by Heiko Oberdiek to allow material to be attached to the location.

```
5034 \def\@glshypernumber#1\nohyperpage#2#3\@nil{%
```

```

5035 \ifx\#1\%
5036 \else
5037 \@delimR#1\delimR\delimR\%
5038 \fi
5039 \ifx\#2\%
5040 \else
5041 #2%
5042 \fi
5043 \ifx\#3\%
5044 \else
5045 \@glsnumber#3\@nil
5046 \fi
5047 }

```

\@delimR displays a range of numbers for the counter whose name is given by \@gls@counter (which must be set prior to using \@glsnumber).

```

\@delimR
5048 \def\@delimR#1\delimR #2\delimR #3\{%
5049 \ifx\#2\%
5050 \@delimN{#1}%
5051 \else
5052 \@gls@numberlink{#1}\delimR\@gls@numberlink{#2}%
5053 \fi}

```

\@delimN displays a list of individual numbers, instead of a range:

```

\@delimN
5054 \def\@delimN#1{\@delimN#1\delimN \delimN\}
5055 \def\@delimN#1\delimN #2\delimN#3\{%
5056 \ifx\#3\%
5057 \@gls@numberlink{#1}%
5058 \else
5059 \@gls@numberlink{#1}\delimN\@gls@numberlink{#2}%
5060 \fi
5061 }

```

The following code is modified from hyperref's \HyInd@pagelink where the name of the counter being used is given by \@gls@counter.

```

5062 \def\@gls@numberlink#1{%
5063 \begingroup
5064 \toks@={}%
5065 \@gls@removespaces#1 \@nil
5066 \endgroup}

5067 \def\@gls@removespaces#1 #2\@nil{%
5068 \toks@=\expandafter{\the\toks@#1}%
5069 \ifx\#2\%
5070 \edef\x{\the\toks@}%
5071 \ifx\x\empty
5072 \else

```

```

5073     \hyperlink{\glentrycounter\glo@counterprefix\the\toks@}%
5074         {\the\toks@}%
5075     \fi
5076 \else
5077     \@gls@ReturnAfterFi{%
5078         \@gls@removespaces#2\@nil
5079     }%
5080 \fi
5081 }
5082 \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`

```
5083 \newcommand*\hyperrm[1]{\textrm{\glshypernumber{#1}}}
```

`\hypersf`

```
5084 \newcommand*\hypersf[1]{\textsf{\glshypernumber{#1}}}
```

`\hypertt`

```
5085 \newcommand*\hypertt[1]{\texttt{\glshypernumber{#1}}}
```

`\hyperbf`

```
5086 \newcommand*\hyperbf[1]{\textbf{\glshypernumber{#1}}}
```

`\hypermd`

```
5087 \newcommand*\hypermd[1]{\textmd{\glshypernumber{#1}}}
```

`\hyperit`

```
5088 \newcommand*\hyperit[1]{\textit{\glshypernumber{#1}}}
```

`\hypersl`

```
5089 \newcommand*\hypersl[1]{\textsl{\glshypernumber{#1}}}
```

`\hyperup`

```
5090 \newcommand*\hyperup[1]{\textup{\glshypernumber{#1}}}
```

`\hypersc`

```
5091 \newcommand*\hypersc[1]{\textsc{\glshypernumber{#1}}}
```

`\hyperemph`

```
5092 \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.

```

5093 \newcommand{\oldacronym}[4][\gls@label]{%
5094   \def\gls@label{#2}%
5095   \newacronym[#4]{#1}{#2}{#3}%
5096   \ifcsundef{xspace}%
5097   {%
5098     \expandafter\edef\csname#1\endcsname{%
5099       \noexpand\@ifstar{\noexpand\Gls{#1}}{\noexpand\gls{#1}}}%
5100   }%
5101 }%
5102 {%
5103   \expandafter\edef\csname#1\endcsname{%
5104     \noexpand\@ifstar{\noexpand\Gls{#1}\noexpand\xspace}{%
5105       \noexpand\gls{#1}\noexpand\xspace}%
5106   }%
5107 }%
5108 }
```

`\newacronym[⟨key-val list⟩]{⟨label⟩}{⟨abbrv⟩}{⟨long⟩}`

This is a quick way of defining acronyms, all it does is call `\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`

```
5109 \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.

```
5110 \newcommand*{\acrpluralsuffix}{\glspluralsuffix}
```

If `garamondx` has been loaded, need to use `\textulc` instead of `\textup`.

`\glstextup`

```
5111 \newrobustcmd*{\glstextup}[1]{\ifdef\textulc{\textulc{#1}}{\textup{#1}}}
```

The following are defined for compatibility with version 2.07 and earlier.

`\glsshortkey`

```
5112 \newcommand*{\glsshortkey}{short}
```

`\glsshortpluralkey`

```
5113 \newcommand*{\glsshortpluralkey}{shortplural}
```

`\glslongkey`

```
5114 \newcommand*{\glslongkey}{long}
```

`\glslongpluralkey`

```
5115 \newcommand*{\glslongpluralkey}{longplural}
```

`\acrfull` Full form of the acronym.

```
5116 \newrobustcmd*{\acrfull}{%
```

```
5117 \@ifstar\s@acrfull\ns@acrfull
```

```
5118 }
```

```
5119 \newcommand*\s@acrfull[2] [] {%
```

```
5120 \new@ifnextchar[{\@acrfull{hyper=false,#1}{#2}}%
```

```
5121 {\@acrfull{hyper=false,#1}{#2} [] }%
```

```
5122 }
```

```
5123 \newcommand*\ns@acrfull[2] [] {%
```

```
5124 \new@ifnextchar[{\@acrfull{#1}{#2}}%
```

```
5125 {\@acrfull{#1}{#2} [] }%
```

```
5126 }
```

`\@acrfull` Low-level macro:

```
5127 \def\@acrfull#1#2[#3]{%
```

Make it easier for acronym styles to change this:

```
5128 \acrfullfmt{#1}{#2}{#3}%  
5129 }
```

Using `\acrlinkfullformat` and `\acrfullformat` is now deprecated as it can cause complications with the first letter upper case variants, but the package needs to provide backward compatibility support.

`\acrfullfmt` No case change full format.

```
5130 \newcommand*{\acrfullfmt}[3]{%  
5131 \acrlinkfullformat{\@acrlong}{\@acrshort}{#1}{#2}{#3}%  
5132 }
```

`\acrlinkfullformat` Format for full links like `\acrfull`. Syntax: `\acrlinkfullformat{<long cs>}{<short cs>}{<options>}{<label>}{<insert>}`

```
5133 \newcommand{\acrlinkfullformat}[5]{%  
5134 \acrfullformat{#1}{#3}{#4}[#5]}{#2}{#3}{#4}[]}%  
5135 }
```

`\acrfullformat` Default full form is `<long>` (`<short>`).

```
5136 \newcommand{\acrfullformat}[2]{#1\space(#2)}
```

Default format for full acronym

`\Acrfull`

```
5137 \newrobustcmd*{\Acrfull}{%  
5138 \@ifstar\s@Acrfull\@ns@Acrfull  
5139 }  
  
5140 \newcommand*\s@Acrfull[2][]{%  
5141 \new@ifnextchar[{\@Acrfull{hyper=false,#1}{#2}}%  
5142 {\@Acrfull{hyper=false,#1}{#2}[]}%  
5143 }  
5144 \newcommand*\@ns@Acrfull[2][]{%  
5145 \new@ifnextchar[{\@Acrfull{#1}{#2}}%  
5146 {\@Acrfull{#1}{#2}[]}%  
5147 }
```

Low-level macro:

```
5148 \def\@Acrfull#1#2[#3]{%
```

Make it easier for acronym styles to change this:

```
5149 \Acrfullfmt{#1}{#2}{#3}%  
5150 }
```

`\Acrfullfmt` First letter upper case full format.

```
5151 \newcommand*{\Acrfullfmt}[3]{%  
5152 \acrlinkfullformat{\@Acrlong}{\@acrshort}{#1}{#2}{#3}%  
5153 }
```

\ACRfull

```
5154 \newrobustcmd*{\ACRfull}{%
5155   \@ifstar\s@ACRfull\ns@ACRfull
5156 }

5157 \newcommand*\s@ACRfull[2][]{%
5158   \new@ifnextchar[{\@ACRfull{hyper=false,#1}{#2}}%
5159     {\@ACRfull{hyper=false,#1}{#2}[]}%
5160 }
5161 \newcommand*\ns@ACRfull[2][]{%
5162   \new@ifnextchar[{\@ACRfull{#1}{#2}}%
5163     {\@ACRfull{#1}{#2}[]}%
5164 }
```

Low-level macro:

```
5165 \def\@ACRfull#1#2[#3]{%
  Make it easier for acronym styles to change this:
5166   \ACRfullfmt{#1}{#2}{#3}%
5167 }
```

\ACRfullfmt All upper case full format.

```
5168 \newcommand*\ACRfullfmt[3]{%
5169   \acrlinkfullformat{\@ACRlong}{\@ACRshort}{#1}{#2}{#3}%
5170 }
```

Plural:

\acrfullpl

```
5171 \newrobustcmd*{\acrfullpl}{%
5172   \@ifstar\s@acrfullpl\ns@acrfullpl
5173 }

5174 \newcommand*\s@acrfullpl[2][]{%
5175   \new@ifnextchar[{\@acrfullpl{hyper=false,#1}{#2}}%
5176     {\@acrfullpl{hyper=false,#1}{#2}[]}%
5177 }
5178 \newcommand*\ns@acrfullpl[2][]{%
5179   \new@ifnextchar[{\@acrfullpl{#1}{#2}}%
5180     {\@acrfullpl{#1}{#2}[]}%
5181 }
```

Low-level macro:

```
5182 \def\@acrfullpl#1#2[#3]{%
  Make it easier for acronym styles to change this:
5183   \acrfullplfmt{#1}{#2}{#3}%
5184 }
```

\acrfullplfmt No case change plural full format.

```
5185 \newcommand*\acrfullplfmt}[3]{%
5186   \acrlinkfullformat{\@acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%
5187 }
```

\Acrfullpl

```
5188 \newrobustcmd*\Acrfullpl}{%
5189   \@ifstar\s@Acrfullpl\ns@Acrfullpl
5190 }
```

```
5191 \newcommand*\s@Acrfullpl[2][]{%
5192   \new@ifnextchar[{\@Acrfullpl{hyper=false,#1}{#2}}%
5193     {\@Acrfullpl{hyper=false,#1}{#2}[]}%
5194 }
5195 \newcommand*\ns@Acrfullpl[2][]{%
5196   \new@ifnextchar[{\@Acrfullpl{#1}{#2}}%
5197     {\@Acrfullpl{#1}{#2}[]}%
5198 }
```

Low-level macro:

```
5199 \def\@Acrfullpl#1#2[#3]{%
```

Make it easier for acronym styles to change this:

```
5200   \Acrfullplfmt{#1}{#2}{#3}%
5201 }
```

\Acrfullplfmt First letter upper case plural full format.

```
5202 \newcommand*\Acrfullplfmt}[3]{%
5203   \acrlinkfullformat{\@acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%
5204 }
```

\ACRfullpl

```
5205 \newrobustcmd*\ACRfullpl}{%
5206   \@ifstar\s@ACRfullpl\ns@ACRfullpl
5207 }
```

```
5208 \newcommand*\s@ACRfullpl[2][]{%
5209   \new@ifnextchar[{\@ACRfullpl{hyper=false,#1}{#2}}%
5210     {\@ACRfullpl{hyper=false,#1}{#2}[]}%
5211 }
5212 \newcommand*\ns@ACRfullpl[2][]{%
5213   \new@ifnextchar[{\@ACRfullpl{#1}{#2}}%
5214     {\@ACRfullpl{#1}{#2}[]}%
5215 }
```

Low-level macro:

```
5216 \def\@ACRfullpl#1#2[#3]{%
```

Make it easier for acronym styles to change this:

```
5217   \ACRfullplfmt{#1}{#2}{#3}%
5218 }
```

`\ACRfullplfmt` All upper case plural full format.

```
5219 \newcommand*{\ACRfullplfmt}[3]{%
5220   \acrlinkfullformat{\@ACRlongpl}{\@ACRshortpl}{#1}{#2}{#3}%
5221 }
```

1.17 Predefined acronym styles

`\acronymfont` This is only used with the additional acronym styles:

```
5222 \newcommand{\acronymfont}[1]{#1}
```

`\firstacronymfont` This is only used with the additional acronym styles:

```
5223 \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.

```
5224 \newcommand*{\acrnameformat}[2]{\acronymfont{#1}}
```

Define some tokens used by `\newacronym`:

`\glskeylisttok`

```
5225 \newtoks\glskeylisttok
```

`\glslabeltok`

```
5226 \newtoks\glslabeltok
```

`\glsshorttok`

```
5227 \newtoks\glsshorttok
```

`\glslongtok`

```
5228 \newtoks\glslongtok
```

`\newacronymhook` Provide a hook for `\newacronym`:

```
5229 \newcommand*{\newacronymhook}{}
```

`\SetGenericNewAcronym` New improved version of setting the acronym style.

```
5230 \newcommand*{\SetGenericNewAcronym}{%
5231   \renewcommand{\newacronym}[4][ ]{%
5232     \ifdefempty{\@glsacronymlists}%
5233     {%
5234       \def\@glo@type{\acronymtype}%
5235       \setkeys{glossentry}{##1}%
5236       \DeclareAcronymList{\@glo@type}%
5237     }%
5238   }%
5239   \glskeylisttok{##1}%
5240   \glslabeltok{##2}%
5241   \glsshorttok{##3}%

```

```

5242 \glslongtok{##4}%
5243 \newacronymhook
5244 \protected@edef\@do@newglossaryentry{%
5245 \noexpand\newglossaryentry{\the\glslabeltok}%
5246 {%
5247 type=\acronymtype,%
5248 name={\expandonce{\acronymentry{##2}}},%
5249 sort={\acronymsort{\the\glsshorttok}{\the\glslongtok}},%
5250 text={\the\glsshorttok},%
5251 short={\the\glsshorttok},%
5252 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5253 long={\the\glslongtok},%
5254 longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5255 \GenericAcronymFields,%
5256 \the\glskeylisttok
5257 }%
5258 }%
5259 \@do@newglossaryentry
5260 }%

```

Make sure that \acrfull etc reflects the new style:

```

5261 \renewcommand*\acrfullfmt}[3]{%
5262 \glslink[##1]{##2}{\genacrfullformat{##2}{##3}}}%
5263 \renewcommand*\Acrfullfmt}[3]{%
5264 \glslink[##1]{##2}{\Genacrfullformat{##2}{##3}}}%
5265 \renewcommand*\ACRfullfmt}[3]{%
5266 \glslink[##1]{##2}{%
5267 \mfirstucMakeUppercase{\genacrfullformat{##2}{##3}}}}%
5268 \renewcommand*\acrfullplfmt}[3]{%
5269 \glslink[##1]{##2}{\genplacrfullformat{##2}{##3}}}%
5270 \renewcommand*\Acrfullplfmt}[3]{%
5271 \glslink[##1]{##2}{\Genplacrfullformat{##2}{##3}}}%
5272 \renewcommand*\ACRfullplfmt}[3]{%
5273 \glslink[##1]{##2}{%
5274 \mfirstucMakeUppercase{\genplacrfullformat{##2}{##3}}}}%

```

Make sure that \glsentryfull etc reflects the new style:

```

5275 \renewcommand*\glsentryfull}[1]{\genacrfullformat{##1}{}}%
5276 \renewcommand*\Glsentryfull}[1]{\Genacrfullformat{##1}{}}%
5277 \renewcommand*\glsentryfullpl}[1]{\genplacrfullformat{##1}{}}%
5278 \renewcommand*\Glsentryfullpl}[1]{\Genplacrfullformat{##1}{}}%
5279 }

```

`\GenericAcronymFields` Fields used by `\SetGenericNewAcronym` that can be changed by the acronym style.

```

5280 \newcommand*\GenericAcronymFields{description={\the\glslongtok}}

```

`\acronymentry` `\acronymentry{<label>}`

Display style for the name field in the list of acronyms.

```
5281 \newcommand*{\acronymentry}[1]{\acronymfont{\glsentryshort{#1}}}
```

`\acronymsort` `\acronymsort{<short>}{<long>}`

Default sort format for acronyms.

```
5282 \newcommand*{\acronymsort}[2]{#1}
```

`\setacronymstyle` `\setacronymstyle{<style name>}`

```
5283 \newcommand*{\setacronymstyle}[1]{%
5284   \ifcsundef{@glsacr@dispstyle@#1}
5285   {%
5286     \PackageError{glossaries}{Undefined acronym style ‘#1’}{}%
5287   }%
5288   {%
5289     \ifdefempty{@glsacronymlists}%
5290     {%
5291       \DeclareAcronymList{\acronymtype}%
5292     }%
5293   }%
5294   \SetGenericNewAcronym
5295   \GlsUseAcrStyleDefs{#1}%
5296   \@for\@gls@type:=\@glsacronymlists\do{%
5297     \defglsentryfmt[\@gls@type]{\GlsUseAcrEntryDispStyle{#1}}%
5298   }%
5299 }%
5300 }
```

`\newacronymstyle` `\newacronymstyle{<style name>}{<entry format definition>}{<display definitions>}`

Defines a new acronym style called *<style name>*.

```
5301 \newcommand*{\newacronymstyle}[3]{%
5302   \ifcsdef{@glsacr@dispstyle@#1}%
5303   {%
5304     \PackageError{glossaries}{Acronym style ‘#1’ already exists}{}%
5305   }%
5306   {%
5307     \csdef{@glsacr@dispstyle@#1}{#2}%
5308     \csdef{@glsacr@styledefs@#1}{#3}%
5309   }%
5310 }
```

seAcrEntryDispStyle

```
5311 \newcommand*{\GlsUseAcrEntryDispStyle}[1]{\csuse{@glsacr@dispstyle@#1}}
```

\GlsUseAcrStyleDefs

```
5312 \newcommand*{\GlsUseAcrStyleDefs}[1]{\csuse{@glsacr@styledefs@#1}}
```

Predefined acronym styles:

long-short *<long>* (*<short>*) acronym style.

```
5313 \newacronymstyle{long-short}%
```

```
5314 {%
```

Check for long form in case this is a mixed glossary.

```
5315 \ifglshaslong{\glslabel}{\glsgenacfmt}{\glsgenentryfmt}%
```

```
5316 }%
```

```
5317 {%
```

```
5318 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
```

```
5319 \renewcommand*{\genacrfullformat}[2]{%
```

```
5320 \glsentrylong{##1}##2\space
```

```
5321 (\protect\firstacronymfont{\glsentryshort{##1}})%
```

```
5322 }%
```

```
5323 \renewcommand*{\Genacrfullformat}[2]{%
```

```
5324 \Glsentrylong{##1}##2\space
```

```
5325 (\protect\firstacronymfont{\glsentryshort{##1}})%
```

```
5326 }%
```

```
5327 \renewcommand*{\genplacrfullformat}[2]{%
```

```
5328 \glsentrylongpl{##1}##2\space
```

```
5329 (\protect\firstacronymfont{\glsentryshortpl{##1}})%
```

```
5330 }%
```

```
5331 \renewcommand*{\Genplacrfullformat}[2]{%
```

```
5332 \Glsentrylongpl{##1}##2\space
```

```
5333 (\protect\firstacronymfont{\glsentryshortpl{##1}})%
```

```
5334 }%
```

```
5335 \renewcommand*{\acronymentry}[1]{\acronymfont{\glsentryshort{##1}}}
```

```
5336 \renewcommand*{\acronymsort}[2]{##1}%
```

```
5337 \renewcommand*{\acronymfont}[1]{##1}%
```

```
5338 \renewcommand*{\firstacronymfont}[1]{\acronymfont{##1}}%
```

```
5339 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
```

```
5340 }
```

short-long *<short>* (*<long>*) acronym style.

```
5341 \newacronymstyle{short-long}%
```

```
5342 {%
```

Check for long form in case this is a mixed glossary.

```
5343 \ifglshaslong{\glslabel}{\glsgenacfmt}{\glsgenentryfmt}%
```

```
5344 }%
```

```
5345 {%
```

```
5346 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
```

```
5347 \renewcommand*{\genacrfullformat}[2]{%
```

```

5348 \protect\firstacronymfont{\glentryshort{##1}}##2\space
5349 (\glentrylong{##1})%
5350 }%
5351 \renewcommand*\Genacrfullformat}[2]{%
5352 \protect\firstacronymfont{\glentryshort{##1}}##2\space
5353 (\glentrylong{##1})%
5354 }%
5355 \renewcommand*\genplacrfullformat}[2]{%
5356 \protect\firstacronymfont{\glentryshortpl{##1}}##2\space
5357 (\glentrylongpl{##1})%
5358 }%
5359 \renewcommand*\Genplacrfullformat}[2]{%
5360 \protect\firstacronymfont{\glentryshortpl{##1}}##2\space
5361 (\glentrylongpl{##1})%
5362 }%
5363 \renewcommand*\acronymentry}[1]{\acronymfont{\glentryshort{##1}}}
5364 \renewcommand*\acronymssort}[2]{##1}%
5365 \renewcommand*\acronymfont}[1]{##1}%
5366 \renewcommand*\firstacronymfont}[1]{\acronymfont{##1}}%
5367 \renewcommand*\acrpluralsuffix{\glspluralsuffix}%
5368 }

```

`long-sc-short` *<long>* (\textsc{<short>}) acronym style.

```

5369 \newacronymstyle{long-sc-short}%
5370 {%
5371 \GlsUseAcrEntryDisplayStyle{long-short}%
5372 }%
5373 {%
5374 \GlsUseAcrStyleDefs{long-short}%
5375 \renewcommand{\acronymfont}[1]{\textsc{##1}}%
5376 \renewcommand*\acrpluralsuffix{\glsstextup{\glspluralsuffix}}%
5377 }

```

`long-sm-short` *<long>* (\textsmaller{<short>}) acronym style.

```

5378 \newacronymstyle{long-sm-short}%
5379 {%
5380 \GlsUseAcrEntryDisplayStyle{long-short}%
5381 }%
5382 {%
5383 \GlsUseAcrStyleDefs{long-short}%
5384 \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
5385 \renewcommand*\acrpluralsuffix{\glspluralsuffix}%
5386 }

```

`sc-short-long` *<short>* (\textsc{<long>}) acronym style.

```

5387 \newacronymstyle{sc-short-long}%
5388 {%
5389 \GlsUseAcrEntryDisplayStyle{short-long}%
5390 }%

```

```

5391 {%
5392   \GlsUseAcrStyleDefs{short-long}%
5393   \renewcommand{\acronymfont}[1]{\textsc{##1}}%
5394   \renewcommand*{\acrpluralsuffix}{\glstextup{\glspluralsuffix}}%
5395 }

```

sm-short-long *<short>* (\textsmaller{<long>}) acronym style.

```

5396 \newacronymstyle{sm-short-long}%
5397 {%
5398   \GlsUseAcrEntryDispStyle{short-long}%
5399 }%
5400 {%
5401   \GlsUseAcrStyleDefs{short-long}%
5402   \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
5403   \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
5404 }

```

long-short-desc *<long>* ({<short>}) acronym style that has an accompanying description (which the user needs to supply).

```

5405 \newacronymstyle{long-short-desc}%
5406 {%
5407   \GlsUseAcrEntryDispStyle{long-short}%
5408 }%
5409 {%
5410   \GlsUseAcrStyleDefs{long-short}%
5411   \renewcommand*{\GenericAcronymFields}{}%
5412   \renewcommand*{\acronymsort}[2]{##2}%
5413   \renewcommand*{\acronymentry}[1]{%
5414     \glstentrylong{##1}\space (\acronymfont{\glstentryshort{##1}})}%
5415 }

```

long-sc-short-desc *<long>* (\textsc{<short>}) acronym style that has an accompanying description (which the user needs to supply).

```

5416 \newacronymstyle{long-sc-short-desc}%
5417 {%
5418   \GlsUseAcrEntryDispStyle{long-sc-short}%
5419 }%
5420 {%
5421   \GlsUseAcrStyleDefs{long-sc-short}%
5422   \renewcommand*{\GenericAcronymFields}{}%
5423   \renewcommand*{\acronymsort}[2]{##2}%
5424   \renewcommand*{\acronymentry}[1]{%
5425     \glstentrylong{##1}\space (\textsc{\acronymfont{\glstentryshort{##1}})}%
5426 }

```

long-sm-short-desc *<long>* (\textsmaller{<short>}) acronym style that has an accompanying description (which the user needs to supply).

```

5427 \newacronymstyle{long-sm-short-desc}%

```

```

5428 {%
5429   \GlsUseAcrEntryDisplayStyle{long-sm-short}%
5430 }%
5431 {%
5432   \GlsUseAcrStyleDefs{long-sm-short}%
5433   \renewcommand*{\GenericAcronymFields}{}%
5434   \renewcommand*{\acronymsort}[2]{##2}%
5435   \renewcommand*{\acronymentry}[1]{%
5436     \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
5437 }

```

short-long-desc *<short>* (*{<long>}*) acronym style that has an accompanying description (which the user needs to supply).

```

5438 \newacronymstyle{short-long-desc}%
5439 {%
5440   \GlsUseAcrEntryDisplayStyle{short-long}%
5441 }%
5442 {%
5443   \GlsUseAcrStyleDefs{short-long}%
5444   \renewcommand*{\GenericAcronymFields}{}%
5445   \renewcommand*{\acronymsort}[2]{##2}%
5446   \renewcommand*{\acronymentry}[1]{%
5447     \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
5448 }

```

sc-short-long-desc *<long>* (\textsc{<short>}) acronym style that has an accompanying description (which the user needs to supply).

```

5449 \newacronymstyle{sc-short-long-desc}%
5450 {%
5451   \GlsUseAcrEntryDisplayStyle{sc-short-long}%
5452 }%
5453 {%
5454   \GlsUseAcrStyleDefs{sc-short-long}%
5455   \renewcommand*{\GenericAcronymFields}{}%
5456   \renewcommand*{\acronymsort}[2]{##2}%
5457   \renewcommand*{\acronymentry}[1]{%
5458     \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
5459 }

```

sm-short-long-desc *<long>* (\textsmaller{<short>}) acronym style that has an accompanying description (which the user needs to supply).

```

5460 \newacronymstyle{sm-short-long-desc}%
5461 {%
5462   \GlsUseAcrEntryDisplayStyle{sm-short-long}%
5463 }%
5464 {%
5465   \GlsUseAcrStyleDefs{sm-short-long}%
5466   \renewcommand*{\GenericAcronymFields}{}%
5467   \renewcommand*{\acronymsort}[2]{##2}%

```

```

5468 \renewcommand*{\acronymentry}[1]{%
5469 \glsentrylong{##1}\space (\acronymfont{\glsentryshort{##1}})}%
5470 }

```

dua <long> only acronym style.

```

5471 \newacronymstyle{dua}%
5472 {%

```

Check for long form in case this is a mixed glossary.

```

5473 \ifdefempty\glscustomtext
5474 {%
5475 \ifglshaslong{\glslabel}%
5476 {%
5477 \glsifplural
5478 {%

```

Plural form:

```

5479 \glscapscase
5480 {%

```

Plural form, don't adjust case:

```

5481 \glsentrylongpl{\glslabel}\glsinsert
5482 }%
5483 {%

```

Plural form, make first letter upper case:

```

5484 \Glsentrylongpl{\glslabel}\glsinsert
5485 }%
5486 {%

```

Plural form, all caps:

```

5487 \mfirstucMakeUppercase
5488 {\glsentrylongpl{\glslabel}\glsinsert}%
5489 }%
5490 }%
5491 {%

```

Singular form

```

5492 \glscapscase
5493 {%

```

Singular form, don't adjust case:

```

5494 \glsentrylong{\glslabel}\glsinsert
5495 }%
5496 {%

```

Subsequent singular form, make first letter upper case:

```

5497 \Glsentrylong{\glslabel}\glsinsert
5498 }%
5499 {%

```

Subsequent singular form, all caps:

```

5500         \mfirstucMakeUppercase
5501         {\glsentrylong{\glslabel}\glsinsert}%
5502     }%
5503 }%
5504 }%
5505 {%

```

Not an acronym:

```

5506     \glsgenentryfmt
5507 }%
5508 }%
5509 {\glscustomtext\glsinsert}%
5510}%
5511{%
5512 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
5513 \renewcommand*{\acrfullfmt}[3]{%
5514     \glslink[##1]{##2}{\glsentryshort{##2}##3\space
5515         (\acronymfont{\glsentryshort{##2}})}}%
5516 \renewcommand*{\Acrfullfmt}[3]{%
5517     \glslink[##1]{##2}{\Glsentrylong{##2}##3\space
5518         (\acronymfont{\glsentryshort{##2}})}}%
5519 \renewcommand*{\ACRfullfmt}[3]{%
5520     \glslink[##1]{##2}{%
5521         \mfirstucMakeUppercase{\glsentrylong{##2}##3\space
5522         (\acronymfont{\glsentryshort{##2}})}}}%
5523 \renewcommand*{\acrfullplfmt}[3]{%
5524     \glslink[##1]{##2}{\glsentryshortpl{##2}##3\space
5525         (\acronymfont{\glsentryshortpl{##2}})}}%
5526 \renewcommand*{\Acrfullplfmt}[3]{%
5527     \glslink[##1]{##2}{\Glsentryshortpl{##2}##3\space
5528         (\acronymfont{\glsentryshortpl{##2}})}}%
5529 \renewcommand*{\ACRfullplfmt}[3]{%
5530     \glslink[##1]{##2}{%
5531         \mfirstucMakeUppercase{\glsentrylongpl{##2}##3\space
5532         (\acronymfont{\glsentryshortpl{##2}})}}}%
5533 \renewcommand*{\glsentryfull}[1]{%
5534     \glsentrylong{##1}\space(\acronymfont{\glsentryshort{##1}})%
5535 }%
5536 \renewcommand*{\Glsentryfull}[1]{%
5537     \Glsentrylong{##1}\space(\acronymfont{\glsentryshort{##1}})%
5538 }%
5539 \renewcommand*{\glsentryfullpl}[1]{%
5540     \glsentrylongpl{##1}\space(\acronymfont{\glsentryshortpl{##1}})%
5541 }%
5542 \renewcommand*{\Glsentryfullpl}[1]{%
5543     \Glsentrylongpl{##1}\space(\acronymfont{\glsentryshortpl{##1}})%
5544 }%
5545 \renewcommand*{\acronymentry}[1]{\acronymfont{\glsentryshort{##1}}}%
5546 \renewcommand*{\acronymsort}[2]{##1}%

```

```

5547 \renewcommand*{\acronymfont}[1]{##1}%
5548 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
5549 }

```

dua-desc *<long>* only acronym style with user-supplied description.

```

5550 \newacronymstyle{dua-desc}%
5551 {%
5552 \GlsUseAcrEntryDispStyle{dua}%
5553 }%
5554 {%
5555 \GlsUseAcrStyleDefs{dua}%
5556 \renewcommand*{\GenericAcronymFields}{}%
5557 \renewcommand*{\acronymentry}[1]{\acronymfont{\glsentrylong{##1}}}
5558 \renewcommand*{\acronymsort}[2]{##2}%
5559 }%

```

footnote *<short>*\footnote{*<long>*} acronym style.

```

5560 \newacronymstyle{footnote}%
5561 {%

```

Check for long form in case this is a mixed glossary.

```

5562 \ifglshaslong{\glslabel}{\glsngenacfmt}{\glsngenentryfmt}%
5563 }%
5564 {%
5565 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%

```

Need to ensure hyperlinks are switched off on first use:

```

5566 \glshyperfirstfalse
5567 \renewcommand*{\genacrfullformat}[2]{%
5568 \protect\firstacronymfont{\glsentryshort{##1}}##2%
5569 \protect\footnote{\glsentrylong{##1}}%
5570 }%
5571 \renewcommand*{\Genacrfullformat}[2]{%
5572 \firstacronymfont{\Glsentryshort{##1}}##2%
5573 \protect\footnote{\glsentrylong{##1}}%
5574 }%
5575 \renewcommand*{\genplacrfullformat}[2]{%
5576 \protect\firstacronymfont{\glsentryshortpl{##1}}##2%
5577 \protect\footnote{\glsentrylongpl{##1}}%
5578 }%
5579 \renewcommand*{\Genplacrfullformat}[2]{%
5580 \protect\firstacronymfont{\Glsentryshortpl{##1}}##2%
5581 \protect\footnote{\glsentrylongpl{##1}}%
5582 }%
5583 \renewcommand*{\acronymentry}[1]{\acronymfont{\glsentryshort{##1}}}%
5584 \renewcommand*{\acronymsort}[2]{##1}%
5585 \renewcommand*{\acronymfont}[1]{##1}%
5586 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%

```

Don't use footnotes for \acrfull:

```

5587 \renewcommand*{\acrfullfmt}[3]{%
5588   \glslink{##1}{##2}{\acronymfont{\glsentryshort{##2}}##3\space
5589   (\glsentrylong{##2})}%
5590 \renewcommand*{\Acrfullfmt}[3]{%
5591   \glslink{##1}{##2}{\acronymfont{\Glsentryshort{##2}}##3\space
5592   (\glsentrylong{##2})}%
5593 \renewcommand*{\ACRfullfmt}[3]{%
5594   \glslink{##1}{##2}{%
5595     \mfirstucMakeUppercase{\acronymfont{\glsentryshort{##2}}##3\space
5596     (\glsentrylong{##2})}%
5597 \renewcommand*{\acrfullplfmt}[3]{%
5598   \glslink{##1}{##2}{\acronymfont{\glsentryshortpl{##2}}##3\space
5599   (\glsentrylongpl{##2})}%
5600 \renewcommand*{\Acrfullplfmt}[3]{%
5601   \glslink{##1}{##2}{\acronymfont{\Glsentryshortpl{##2}}##3\space
5602   (\glsentrylongpl{##2})}%
5603 \renewcommand*{\ACRfullplfmt}[3]{%
5604   \glslink{##1}{##2}{%
5605     \mfirstucMakeUppercase{\acronymfont{\glsentryshortpl{##2}}##3\space
5606     (\glsentrylongpl{##2})}%

```

Similarly for \glsentryfull etc:

```

5607 \renewcommand*{\glsentryfull}[1]{%
5608   \acronymfont{\glsentryshort{##1}}\space(\glsentrylong{##1})}%
5609 \renewcommand*{\Glsentryfull}[1]{%
5610   \acronymfont{\Glsentryshort{##1}}\space(\glsentrylong{##1})}%
5611 \renewcommand*{\glsentryfullpl}[1]{%
5612   \acronymfont{\glsentryshortpl{##1}}\space(\glsentrylongpl{##1})}%
5613 \renewcommand*{\Glsentryfullpl}[1]{%
5614   \acronymfont{\Glsentryshortpl{##1}}\space(\glsentrylongpl{##1})}%
5615 }

```

footnote-sc \textsc{<short>}\footnote{<long>} acronym style.

```

5616 \newacronymstyle{footnote-sc}%
5617 {%
5618   \GlsUseAcrEntryDispStyle{footnote}%
5619 }%
5620 {%
5621   \GlsUseAcrStyleDefs{footnote}%
5622 \renewcommand{\acronymentry}[1]{\acronymfont{\glsentryshort{##1}}}
5623 \renewcommand{\acronymfont}[1]{\textsc{##1}}%
5624 \renewcommand*{\acrpluralsuffix}{\glstextup{\glspluralsuffix}}%
5625 }%

```

footnote-sm \textsmaller{<short>}\footnote{<long>} acronym style.

```

5626 \newacronymstyle{footnote-sm}%
5627 {%
5628   \GlsUseAcrEntryDispStyle{footnote}%
5629 }%
5630 {%

```

```

5631 \GlsUseAcrStyleDefs{footnote}%
5632 \renewcommand{\acronymentry}[1]{\acronymfont{\glentryshort{##1}}}
5633 \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
5634 \renewcommand*{\acrpluralsuffix}{\glpluralsuffix}%
5635 }%

```

footnote-desc *<short>*\footnote{*<long>*} acronym style that has an accompanying description (which the user needs to supply).

```

5636 \newacronymstyle{footnote-desc}%
5637 {%
5638 \GlsUseAcrEntryDisplayStyle{footnote}%
5639 }%
5640 {%
5641 \GlsUseAcrStyleDefs{footnote}%
5642 \renewcommand*{\GenericAcronymFields}{}%
5643 \renewcommand*{\acronymsort}[2]{##2}%
5644 \renewcommand*{\acronymentry}[1]{%
5645 \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
5646 }

```

footnote-sc-desc \textsc{<short>}\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```

5647 \newacronymstyle{footnote-sc-desc}%
5648 {%
5649 \GlsUseAcrEntryDisplayStyle{footnote-sc}%
5650 }%
5651 {%
5652 \GlsUseAcrStyleDefs{footnote-sc}%
5653 \renewcommand*{\GenericAcronymFields}{}%
5654 \renewcommand*{\acronymsort}[2]{##2}%
5655 \renewcommand*{\acronymentry}[1]{%
5656 \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
5657 }

```

footnote-sm-desc \textsmaller{<short>}\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```

5658 \newacronymstyle{footnote-sm-desc}%
5659 {%
5660 \GlsUseAcrEntryDisplayStyle{footnote-sm}%
5661 }%
5662 {%
5663 \GlsUseAcrStyleDefs{footnote-sm}%
5664 \renewcommand*{\GenericAcronymFields}{}%
5665 \renewcommand*{\acronymsort}[2]{##2}%
5666 \renewcommand*{\acronymentry}[1]{%
5667 \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
5668 }

```

fineAcronymSynonyms

5669 \newcommand*{\DefineAcronymSynonyms}{%

Short form

\acs

5670 \let\acs\acrshort

First letter uppercase short form

\Acs

5671 \let\Acs\Acrshort

Plural short form

\acsp

5672 \let\acsp\acrshortpl

First letter uppercase plural short form

\Acsp

5673 \let\Acsp\Acrshortpl

Long form

\acl

5674 \let\acl\aclong

Plural long form

\aclp

5675 \let\aclp\aclongpl

First letter upper case long form

\Acl

5676 \let\Acl\Aclong

First letter upper case plural long form

\Aclp

5677 \let\Aclp\Aclongpl

Full form

\acf

5678 \let\acf\acrfull

Plural full form

\acfp

5679 \let\acfp\acrfullpl

First letter upper case full form

`\Acf`

```
5680 \let\Acf\Acrfull
```

First letter upper case plural full form

`\Acfp`

```
5681 \let\Acfp\Acrfullpl
```

Standard form

`\ac`

```
5682 \let\ac\gls
```

First upper case standard form

`\Ac`

```
5683 \let\Ac\Gls
```

Standard plural form

`\acp`

```
5684 \let\acp\glspl
```

Standard first letter upper case plural form

`\Acp`

```
5685 \let\Acp\Glspl
```

```
5686 }
```

Define synonyms if required

```
5687 \ifglsacrshortcuts
```

```
5688 \DefineAcronymSynonyms
```

```
5689 \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.

```
5690 \newcommand*\SetDefaultAcronymDisplayStyle}[1]{%
```

```
5691 \defglsentryfmt[#1]{\glsentryfmt}%
```

```
5692 }
```

defaultNewAcronymDef Sets up the acronym definition for the default style. The information is provided by the tokens `\glslabeltok`, `\glsshorttok`, `\glslongtok` and `\glskeylisttok`.

```
5693 \newcommand*\DefaultNewAcronymDef{%
```

```
5694 \edef\@do@newglossaryentry{%
```

```
5695 \noexpand\newglossaryentry{\the\glslabeltok}%
```

```
5696 {%
```

```

5697     type=\acronymtype,%
5698     name={\the\glsshorttok},%
5699     sort={\the\glsshorttok},%
5700     text={\the\glsshorttok},%
5701     first={\acrfullformat{\the\glslongtok}{\the\glsshorttok}},%
5702     plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5703     firstplural={\acrfullformat{\noexpand\expandonce\noexpand\@glo@longpl}%
5704                 {\noexpand\expandonce\noexpand\@glo@shortpl}},%
5705     short={\the\glsshorttok},%
5706     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5707     long={\the\glslongtok},%
5708     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5709     description={\the\glslongtok},%
5710     descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%

```

Remaining options specified by the user:

```

5711     \the\glskeylisttok
5712 }%
5713 }%
5714 \let\@org@gls@assign@firstpl\gls@assign@firstpl
5715 \let\@org@gls@assign@plural\gls@assign@plural
5716 \let\@org@gls@assign@descplural\gls@assign@descplural
5717 \def\gls@assign@firstpl##1##2{%
5718     \@@gls@expand@field{##1}{firstpl}{##2}%
5719 }%
5720 \def\gls@assign@plural##1##2{%
5721     \@@gls@expand@field{##1}{plural}{##2}%
5722 }%
5723 \def\gls@assign@descplural##1##2{%
5724     \@@gls@expand@field{##1}{descplural}{##2}%
5725 }%
5726 \do@newglossaryentry
5727 \let\gls@assign@firstpl\@org@gls@assign@firstpl
5728 \let\gls@assign@plural\@org@gls@assign@plural
5729 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
5730 }

```

DefaultAcronymStyle Set up the default acronym style:

```

5731 \newcommand*\SetDefaultAcronymStyle{%

```

Set the display style:

```

5732     \@for\@gls@type:=\@glsacronymlists\do{%
5733         \SetDefaultAcronymDisplayStyle{\@gls@type}%
5734     }%

```

Set up the definition of \newacronym:

```

5735 \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).

```

5736 \ifx\@glsacronymlists\@empty
5737 \def\@glo@type{\acronymtype}%
5738 \setkeys{glossentry}{##1}%
5739 \DeclareAcronymList{\@glo@type}%
5740 \SetDefaultAcronymDisplayStyle{\@glo@type}%
5741 \fi
5742 \glskeylisttok{##1}%
5743 \glslabeltok{##2}%
5744 \glsshorttok{##3}%
5745 \glslongtok{##4}%
5746 \newacronymhook
5747 \DefaultNewAcronymDef
5748 }%
5749 \renewcommand*\@acrpluralsuffix{\@glspluralsuffix}%
5750 }

```

`\acrfootnote` Used by the footnote acronym styles.

```

5751 \newcommand*\acrfootnote[3]{\acrlinkfootnote{#1}{#2}{#3}}

```

`\acrlinkfootnote`

```

5752 \newcommand*\acrlinkfootnote[3]{%
5753 \footnote{\glslink[#1]{#2}{#3}}%
5754 }

```

`\acrnoflinkfootnote`

```

5755 \newcommand*\acrnoflinkfootnote[3]{%
5756 \footnote{#3}%
5757 }

```

`AcronymDisplayStyle` Sets the acronym display style for given glossary for the description and footnote combination.

```

5758 \newcommand*\SetDescriptionFootnoteAcronymDisplayStyle[1]{%
5759 \def\glsentryfmt[1]{%

```

```

5760 \ifdefempty\glsustomtext
5761 {%
5762 \ifglsused{\glslabel}%
5763 {%
5764 \acronymfont{\glsentryfmt}%
5765 }%
5766 {%
5767 \firstacronymfont{\glsentryfmt}%
5768 \ifglsymbol{\glslabel}%
5769 {%
5770 \expandafter\protect\expandafter\acrfootnote\expandafter
5771 {\@gls@link@opts}{\@gls@link@label}%
5772 }%
5773 \glsifplural
5774 {\glsentrysymbolplural{\glslabel}}%

```

```

5775         {\glsentrysymbol{\glslabel}}}%
5776     }%
5777 }%
5778 }%
5779 }%
5780 {\glscustomtext\glsinsert}%
5781 }%
5782 }

```

otnoteNewAcronymDef

```

5783 \newcommand*{\DescriptionFootnoteNewAcronymDef}{%
5784   \edef\@do@newglossaryentry{%
5785     \noexpand\newglossaryentry{\the\glslabeltok}%
5786     {%
5787       type=\acronymtype,%
5788       name={\noexpand\acronymfont{\the\glsshorttok}},%
5789       sort={\the\glsshorttok},%
5790       first={\the\glsshorttok},%
5791       firstplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5792       text={\the\glsshorttok},%
5793       plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5794       short={\the\glsshorttok},%
5795       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5796       long={\the\glslongtok},%
5797       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5798       symbol={\the\glslongtok},%
5799       symbolplural={\noexpand\expandonce\noexpand\@glo@longpl},%
5800       \the\glskeylisttok
5801     }%
5802   }%
5803   \let\@org@gls@assign@firstpl\gls@assign@firstpl
5804   \let\@org@gls@assign@plural\gls@assign@plural
5805   \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
5806   \def\gls@assign@firstpl##1##2{%
5807     \@@gls@expand@field{##1}{firstpl}{##2}%
5808   }%
5809   \def\gls@assign@plural##1##2{%
5810     \@@gls@expand@field{##1}{plural}{##2}%
5811   }%
5812   \def\gls@assign@symbolplural##1##2{%
5813     \@@gls@expand@field{##1}{symbolplural}{##2}%
5814   }%
5815   \@do@newglossaryentry
5816   \let\gls@assign@plural\@org@gls@assign@plural
5817   \let\gls@assign@firstpl\@org@gls@assign@firstpl
5818   \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
5819 }

```

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.

```

5820 \newcommand*{\SetDescriptionFootnoteAcronymStyle}{%
5821   \renewcommand{\newacronym}[4][]{}%
5822   \ifx\@glsacronymlists\@empty
5823     \def\@glo@type{\acronymtype}%
5824     \setkeys{glossentry}{##1}%
5825     \DeclareAcronymList{\@glo@type}%
5826     \SetDescriptionFootnoteAcronymDisplayStyle{\@glo@type}%
5827   \fi
5828   \glskeylisttok{##1}%
5829   \glslabeltok{##2}%
5830   \glsshorttok{##3}%
5831   \glslongtok{##4}%
5832   \newacronymhook
5833   \DescriptionFootnoteNewAcronymDef
5834 }%
```

If footnote package option is specified, set the first use to append the long form (stored in symbol) as a footnote.

```

5835 \@for\@gls@type:=\@glsacronymlists\do{%
5836   \SetDescriptionFootnoteAcronymDisplayStyle{\@gls@type}%
5837 }%
```

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.

```

5838 \ifglsacrsmallcaps
5839   \renewcommand*{\acronymfont}[1]{\textsc{##1}}%
5840   \renewcommand*{\acrpluralsuffix}{%
5841     \glstextup{\glspluralsuffix}}%
5842   \else
5843     \ifglsacrsmaller
5844       \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%
5845     \fi
5846   \fi
```

Check for package option clash

```

5847 \ifglsacrdua
5848   \PackageError{glossaries}{Option clash: ‘footnote’ and ‘dua’
5849     can’t both be set}{}%
5850 \fi
5851 }%
```

AcronymDisplayStyle Sets the acronym display style for given glossary with description and dua combination.

```

5852 \newcommand*{\SetDescriptionDUAacronymDisplayStyle}[1]{%
5853   \defglsentryfmt[#1]{\glsgenentryfmt}%
5854 }
```

tionDUANewAcronymDef

```

5855 \newcommand*{\DescriptionDUANewAcronymDef}{%
5856   \edef\@do@newglossaryentry{%
5857     \noexpand\newglossaryentry{\the\glslabeltok}%
5858     {%
5859       type=\acronymtype,%
5860       name={\the\glslongtok},%
5861       sort={\the\glslongtok},%
5862       text={\the\glslongtok},%
5863       first={\the\glslongtok},%
5864       plural={\noexpand\expandonce\noexpand\@glo@longpl},%
5865       firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
5866       short={\the\glsshorttok},%
5867       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5868       long={\the\glslongtok},%
5869       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5870       symbol={\the\glsshorttok},%
5871       symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5872       \the\glskeylisttok
5873     }%
5874   }%
5875   \let\@org@gls@assign@firstpl\gls@assign@firstpl
5876   \let\@org@gls@assign@plural\gls@assign@plural
5877   \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
5878   \def\gls@assign@firstpl##1##2{%
5879     \@@gls@expand@field{##1}{firstpl}{##2}%
5880   }%
5881   \def\gls@assign@plural##1##2{%
5882     \@@gls@expand@field{##1}{plural}{##2}%
5883   }%
5884   \def\gls@assign@symbolplural##1##2{%
5885     \@@gls@expand@field{##1}{symbolplural}{##2}%
5886   }%
5887   \@do@newglossaryentry
5888   \let\gls@assign@firstpl\@org@gls@assign@firstpl
5889   \let\gls@assign@plural\@org@gls@assign@plural
5890   \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
5891 }

```

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.

```

5892 \newcommand*{\SetDescriptionDUAAcronymStyle}{%
5893   \ifglsacrsmallcaps
5894     \PackageError{glossaries}{Option clash: 'smallcaps' and 'dua'
5895       can't both be set}{}%
5896   \else
5897     \ifglsacrsmaller
5898       \PackageError{glossaries}{Option clash: 'smaller' and 'dua'

```

```

5899     can't both be set}{}%
5900   \fi
5901 \fi
5902 \renewcommand{\newacronym}[4][]{%
5903   \ifx\@glsacronymlists\@empty
5904     \def\@glo@type{\acronymtype}%
5905     \setkeys{glossentry}{##1}%
5906     \DeclareAcronymList{\@glo@type}%
5907     \SetDescriptionDUAAcronymDisplayStyle{\@glo@type}%
5908   \fi
5909   \glskeylisttok{##1}%
5910   \glslabeltok{##2}%
5911   \glsshorttok{##3}%
5912   \glslongtok{##4}%
5913   \newacronymhook
5914   \DescriptionDUANewAcronymDef
5915 }%

Set display.
5916 \@for\@gls@type:=\@glsacronymlists\do{%
5917   \SetDescriptionDUAAcronymDisplayStyle{\@gls@type}%
5918 }%
5919 }%

```

AcronymDisplayStyle Sets the acronym display style for given glossary using the description setting (but not footnote or dua).

```

5920 \newcommand*\SetDescriptionAcronymDisplayStyle[1]{%
5921   \def\glsentryfmt[1]{%

5922     \ifdefempty\glscustomtext
5923     {%
5924       \ifglsused{\glslabel}%
5925       {%

Move the inserted text outside of \acronymfont
5926         \let\gls@org@insert\glsinsert
5927         \let\glsinsert\@empty
5928         \acronymfont{\glsentryfmt}\gls@org@insert
5929       }%
5930     {%
5931       \glsentryfmt
5932       \ifglsymbol{\glslabel}%
5933       {%
5934         \glsifplural
5935         {%
5936           \def\@glo@symbol{\glsentrysymbolplural{\glslabel}}%
5937         }%
5938       {%
5939         \def\@glo@symbol{\glsentrysymbol{\glslabel}}%
5940       }%

```

```

5941         \space(\protect\firstacronymfont
5942         {\glscapscase
5943         {\@glo@symbol}
5944         {\@glo@symbol}
5945         {\mfirstucMakeUppercase{\@glo@symbol}}})%
5946     }%
5947     {}%
5948 }%
5949 }%
5950 {\glscustomtext\glsinsert}%
5951 }%
5952 }

```

ptionNewAcronymDef

```

5953 \newcommand*{\DescriptionNewAcronymDef}{%
5954 \edef\@do@newglossaryentry{%
5955 \noexpand\newglossaryentry{\the\glslabeltok}%
5956 {%
5957     type=\acronymtype,%
5958     name={\noexpand
5959         \acnameformat{\the\glsshorttok}{\the\glslongtok}},%
5960     sort={\the\glsshorttok},%
5961     first={\the\glslongtok},%
5962     firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
5963     text={\the\glsshorttok},%
5964     plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5965     short={\the\glsshorttok},%
5966     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5967     long={\the\glslongtok},%
5968     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5969     symbol={\noexpand\@glo@text},%
5970     symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
5971     \the\glskeylisttok}%
5972 }%
5973 \let\@org@gls@assign@firstpl\gls@assign@firstpl
5974 \let\@org@gls@assign@plural\gls@assign@plural
5975 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
5976 \def\gls@assign@firstpl##1##2{%
5977     \@@gls@expand@field{##1}{firstpl}{##2}%
5978 }%
5979 \def\gls@assign@plural##1##2{%
5980     \@@gls@expand@field{##1}{plural}{##2}%
5981 }%
5982 \def\gls@assign@symbolplural##1##2{%
5983     \@@gls@expand@field{##1}{symbolplural}{##2}%
5984 }%
5985 \@do@newglossaryentry
5986 \let\gls@assign@firstpl\@org@gls@assign@firstpl
5987 \let\gls@assign@plural\@org@gls@assign@plural

```

```

5988 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
5989 }

```

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.

```

5990 \newcommand*{\SetDescriptionAcronymStyle}{%
5991   \renewcommand{\newacronym}[4][\]{%
5992     \ifx\@glsacronymlists\@empty
5993       \def\@glo@type{\acronymtype}%
5994       \setkeys{glossentry}{##1}%
5995       \DeclareAcronymList{\@glo@type}%
5996       \SetDescriptionAcronymDisplayStyle{\@glo@type}%
5997     \fi
5998     \glskeylisttok{##1}%
5999     \glslabeltok{##2}%
6000     \glsshorttok{##3}%
6001     \glslongtok{##4}%
6002     \newacronymhook
6003     \DescriptionNewAcronymDef
6004   }%

```

Set display.

```

6005 \@for\@gls@type:=\@glsacronymlists\do{%
6006   \SetDescriptionAcronymDisplayStyle{\@gls@type}%
6007 }%

```

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.

```

6008 \ifglsacrsmallcaps
6009   \renewcommand{\acronymfont}[1]{\textsc{##1}}
6010   \renewcommand*{\acrpluralsuffix}{%
6011     \glstextup{\glspluralsuffix}}%
6012 \else
6013   \ifglsacrsmaller
6014     \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%
6015   \fi
6016 \fi
6017 }%

```

AcronymDisplayStyle Sets the acronym display style for given glossary with footnote setting (but not description or dua).

```

6018 \newcommand*{\SetFootnoteAcronymDisplayStyle}[1]{%
6019   \defglsentryfmt[#1]{%
6020     \ifdefempty\glscustomtext
6021     {%

```

Move the inserted text outside of \acronymfont

```

6022     \let\gls@org@insert\glsinsert
6023     \let\glsinsert\@empty
6024     \ifglsused{\glslabel}%
6025     {%
6026         \acronymfont{\glsgenentryfmt}\gls@org@insert
6027     }%
6028     {%
6029         \firstacronymfont{\glsgenentryfmt}\gls@org@insert
6030         \ifglsashaslong{\glslabel}%
6031         {%
6032             \expandafter\protect\expandafter\acrfootnote\expandafter
6033             {\@gls@link@opts}{\@gls@link@label}%
6034             {%
6035                 \glsifplural
6036                 {\glsentrylongpl{\glslabel}}%
6037                 {\glsentrylong{\glslabel}}%
6038             }%
6039         }%
6040     }%
6041 }%
6042 }%
6043 {\gls@customtext\glsinsert}%
6044 }%
6045 }

```

otnoteNewAcronymDef

```

6046 \newcommand*{\FootnoteNewAcronymDef}{%
6047     \edef\@do@newglossaryentry{%
6048         \noexpand\newglossaryentry{\the\glslabeltok}%
6049         {%
6050             type=\acronymtype,%
6051             name={\noexpand\acronymfont{\the\glsshorttok}},%
6052             sort={\the\glsshorttok},%
6053             text={\the\glsshorttok},%
6054             plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6055             first={\the\glsshorttok},%
6056             firstplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6057             short={\the\glsshorttok},%
6058             shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6059             long={\the\glslongtok},%
6060             longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6061             description={\the\glslongtok},%
6062             descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6063             \the\glskeylisttok
6064         }%
6065     }%
6066     \let\@org@gls@assign@plural\gls@assign@plural

```

```

6067 \let\@org@gl@s@assign@firstpl\gl@s@assign@firstpl
6068 \let\@org@gl@s@assign@descplural\gl@s@assign@descplural
6069 \def\gl@s@assign@firstpl##1##2{%
6070   \@@gl@s@expand@field{##1}{firstpl}{##2}%
6071 }%
6072 \def\gl@s@assign@plural##1##2{%
6073   \@@gl@s@expand@field{##1}{plural}{##2}%
6074 }%
6075 \def\gl@s@assign@descplural##1##2{%
6076   \@@gl@s@expand@field{##1}{descplural}{##2}%
6077 }%
6078 \do@newglossaryentry
6079 \let\gl@s@assign@plural\@org@gl@s@assign@plural
6080 \let\gl@s@assign@firstpl\@org@gl@s@assign@firstpl
6081 \let\gl@s@assign@descplural\@org@gl@s@assign@descplural
6082 }

```

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.

```

6083 \newcommand*\SetFootnoteAcronymStyle{%
6084   \renewcommand{\newacronym}[4][\]{%
6085     \ifx\@gl@s@acronymlists\@empty
6086       \def\@glo@type{\acronymtype}%
6087       \setkeys{glossentry}{##1}%
6088       \DeclareAcronymList{\@glo@type}%
6089       \SetFootnoteAcronymDisplayStyle{\@glo@type}%
6090     \fi
6091     \gl@keylisttok{##1}%
6092     \gl@labeltok{##2}%
6093     \gl@shorttok{##3}%
6094     \gl@longtok{##4}%
6095     \newacronymhook
6096     \FootnoteNewAcronymDef
6097   }%

```

Set display

```

6098 \@for\@gl@s@type:=\@gl@s@acronymlists\do{%
6099   \SetFootnoteAcronymDisplayStyle{\@gl@s@type}%
6100 }%

```

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.

```

6101 \ifgl@sacrsmallcaps
6102   \renewcommand*\acronymfont}[1]{\textsc{##1}}%
6103   \renewcommand*\acrpluralsuffix{%
6104     \gl@textup{\gl@pluralsuffix}}%
6105 \else
6106   \ifgl@sacrsmaller

```

```

6107      \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%
6108      \fi
6109      \fi

    Check for option clash
6110      \ifglssacrdua
6111          \PackageError{glossaries}{Option clash: ‘footnote’ and ‘dua’
6112              can’t both be set}{}%
6113      \fi
6114 }%

```

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.

```

6115 \DeclareRobustCommand*{\glsdoparenifnotempty}[2]{%
6116     \protected@edef\gls@tmp{#1}%
6117     \ifdefempty\gls@tmp
6118     {%
6119     {%
6120         \ifx\gls@tmp\@gls@default@value
6121         \else
6122             \space (#2{#1})%
6123         \fi
6124     }%
6125 }

```

AcronymDisplayStyle Sets the acronym display style for given glossary where neither footnote nor description is required, but smallcaps or smaller specified.

```

6126 \newcommand*{\SetSmallAcronymDisplayStyle}[1]{%
6127     \defglsenentryfmt[#1]{%

        Move the inserted text outside of \acronymfont
6130         \let\gls@org@insert\glsinsert
6131         \let\glsinsert\@empty
6132         \ifglssused{\glslabel}%
6133         {%
6134             \acronymfont{\glsgenentryfmt}\gls@org@insert
6135         }%
6136         {%
6137             \glsgenentryfmt
6138             \ifglshassymbol{\glslabel}%
6139             {%
6140                 \glsifplural
6141                 {%
6142                     \def\@gls@symbol{\glsentrysymbolplural{\glslabel}}%
6143                 }%
6144                 {%

```

```

6145         \def\@glo@symbol{\glsentrysymbol{\glslabel}}%
6146     }%
6147     \space
6148     (\glscapscase
6149     {\firstacronymfont{\@glo@symbol}}%
6150     {\firstacronymfont{\@glo@symbol}}%
6151     {\firstacronymfont{\mfirstucMakeUppercase{\@glo@symbol}}})%
6152 }%
6153 {}%
6154 }%
6155 }%
6156 {\glscustomtext\glsinsert}%
6157 }%
6158 }

```

\SmallNewAcronymDef

```

6159 \newcommand*{\SmallNewAcronymDef}{%
6160     \edef\@do@newglossaryentry{%
6161         \noexpand\newglossaryentry{\the\glslabeltok}%
6162         {%
6163             type=\acronymtype,%
6164             name={\noexpand\acronymfont{\the\glsshorttok}},%
6165             sort={\the\glsshorttok},%
6166             text={\the\glsshorttok},%

```

Default to the short plural.

```

6167     plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6168     first={\the\glslongtok},%

```

Default to the long plural.

```

6169     firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6170     short={\the\glsshorttok},%
6171     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6172     long={\the\glslongtok},%
6173     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6174     description={\noexpand\@glo@first},%
6175     descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6176     symbol={\the\glsshorttok},%

```

Default to the short plural.

```

6177     symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6178     \the\glskeylisttok
6179 }%
6180 }%
6181 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6182 \let\@org@gls@assign@plural\gls@assign@plural
6183 \let\@org@gls@assign@descplural\gls@assign@descplural
6184 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6185 \def\gls@assign@firstpl##1##2{%
6186     \@@gls@expand@field{##1}{firstpl}{##2}%

```

```

6187 }%
6188 \def\gls@assign@plural##1##2{%
6189   \@@gls@expand@field{##1}{plural}{##2}%
6190 }%
6191 \def\gls@assign@descplural##1##2{%
6192   \@@gls@expand@field{##1}{descplural}{##2}%
6193 }%
6194 \def\gls@assign@symbolplural##1##2{%
6195   \@@gls@expand@field{##1}{symbolplural}{##2}%
6196 }%
6197 \do@newglossaryentry
6198 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6199 \let\gls@assign@plural\@org@gls@assign@plural
6200 \let\gls@assign@descplural\@org@gls@assign@descplural
6201 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6202 }

```

`\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.

```

6203 \newcommand*{\SetSmallAcronymStyle}{%
6204   \renewcommand{\newacronym}[4][\]{%
6205     \ifx\@glsacronymlists\@empty
6206       \def\@glo@type{\acronymtype}%
6207       \setkeys{glossentry}{##1}%
6208       \DeclareAcronymList{\@glo@type}%
6209       \SetSmallAcronymDisplayStyle{\@glo@type}%
6210     \fi
6211     \glskeylisttok{##1}%
6212     \glslabeltok{##2}%
6213     \glsshorttok{##3}%
6214     \glslongtok{##4}%
6215     \newacronymhook
6216     \SmallNewAcronymDef
6217   }%

```

Change the display since first only contains long form.

```

6218   \@for\@gls@type:=\@glsacronymlists\do{%
6219     \SetSmallAcronymDisplayStyle{\@gls@type}%
6220   }%

```

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.

```

6221   \ifglsacrsmallcaps
6222     \renewcommand*{\acronymfont}[1]{\textsc{##1}}
6223     \renewcommand*{\acrpluralsuffix}{%
6224       \glstextup{\glspluralsuffix}}%
6225   \else
6226     \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}
6227   \fi

```

check for option clash

```
6228 \ifglSacrdua
6229 \ifglSacrsMallcaps
6230 \PackageError{glossaries}{Option clash: ‘smallcaps’ and ‘dua’
6231 can’t both be set}{}%
6232 \else
6233 \PackageError{glossaries}{Option clash: ‘smaller’ and ‘dua’
6234 can’t both be set}{}%
6235 \fi
6236 \fi
6237 }%
```

`\SetDUADisplayStyle` Sets the acronym display style for given glossary with dua setting.

```
6238 \newcommand*{\SetDUADisplayStyle}[1]{%
6239 \defglSentryfmt[#1]{\glSgenentryfmt}%
6240 }
```

`\DUANewAcronymDef`

```
6241 \newcommand*{\DUANewAcronymDef}{%
6242 \edef\@do@newglossaryentry{%
6243 \noexpand\newglossaryentry{\the\glSlabeltok}%
6244 {%
6245 type=\acronymtype,%
6246 name={\the\glSshorttok},%
6247 text={\the\glSlongtok},%
6248 first={\the\glSlongtok},%
6249 plural={\noexpand\expandonce\noexpand\@glo@longpl},%
6250 firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6251 short={\the\glSshorttok},%
6252 shortplural={\the\glSshorttok\noexpand\acrpluralsuffix},%
6253 long={\the\glSlongtok},%
6254 longplural={\the\glSlongtok\noexpand\acrpluralsuffix},%
6255 description={\the\glSlongtok},%
6256 descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6257 symbol={\the\glSshorttok},%
6258 symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6259 \the\glSkeylisttok
6260 }%
6261 }%
6262 \let\@org@glS@assign@firstpl\glS@assign@firstpl
6263 \let\@org@glS@assign@plural\glS@assign@plural
6264 \let\@org@glS@assign@symbolplural\glS@assign@symbolplural
6265 \let\@org@glS@assign@descplural\glS@assign@descplural
6266 \def\glS@assign@firstpl##1##2{%
6267 \@@glS@expand@field{##1}{firstpl}{##2}%
6268 }%
6269 \def\glS@assign@plural##1##2{%
6270 \@@glS@expand@field{##1}{plural}{##2}%
6271 }%
```

```

6272 \def\gls@assign@symbolplural##1##2{%
6273   \@@gls@expand@field{##1}{symbolplural}{##2}%
6274 }%
6275 \def\gls@assign@descplural##1##2{%
6276   \@@gls@expand@field{##1}{descplural}{##2}%
6277 }%
6278 \do@newglossaryentry
6279 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6280 \let\gls@assign@plural\@org@gls@assign@plural
6281 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6282 \let\gls@assign@descplural\@org@gls@assign@descplural
6283 }

```

`\SetDUASyle` Always expand acronyms.

```

6284 \newcommand*{\SetDUASyle}{%
6285   \renewcommand{\newacronym}[4][]{%
6286     \ifx\@glsacronymlists\@empty
6287       \def\@glo@type{\acronymtype}%
6288       \setkeys{glossentry}{##1}%
6289       \DeclareAcronymList{\@glo@type}%
6290       \SetDUADisplayStyle{\@glo@type}%
6291     \fi
6292     \glskeylisttok{##1}%
6293     \glslabeltok{##2}%
6294     \glsshorttok{##3}%
6295     \glslongtok{##4}%
6296     \newacronymhook
6297     \DUANewAcronymDef
6298   }%
6299   \SetDUADisplayStyle{\@glsacronymlists\do}%
6300   \SetDUADisplayStyle{\@gls@type}%
6301 }%
6302 }

```

Set the display

`\SetAcronymStyle`

```

6303 \newcommand*{\SetAcronymStyle}{%
6304   \SetDefaultAcronymStyle
6305   \ifglsacrdescription
6306     \ifglsacrfootnote
6307       \SetDescriptionFootnoteAcronymStyle
6308     \else
6309       \ifglsacrdua
6310         \SetDescriptionDUAAcronymStyle
6311       \else
6312         \SetDescriptionAcronymStyle
6313       \fi
6314     \fi
6315   \else

```

```

6316 \ifglsacrfootnote
6317 \SetFootnoteAcronymStyle
6318 \else
6319 \ifthenelse{\boolean{glsacrsmallcaps}}\OR
6320 \boolean{glsacrsmaller}}}%
6321 {%
6322 \SetSmallAcronymStyle
6323 }%
6324 {%
6325 \ifglsacrdua
6326 \SetDUASStyle
6327 \fi
6328 }%
6329 \fi
6330 \fi
6331 }

```

Set the acronym style according to the package options

```
6332 \SetAcronymStyle
```

Allow user to define their own custom acronyms. (For compatibility with versions before v3.0, the short form is stored in the user1 key, the plural short form is stored in the user2 key, the long form is stored in the user3 key and the plural long form is stored in the user4 key.) Defaults to displaying only the acronym with the long form as the description.

`\SetCustomDisplayStyle` Sets the acronym display style.

```

6333 \newcommand*{\SetCustomDisplayStyle}[1]{%
6334 \defglsentryfmt[#1]{\glsentryfmt}%
6335 }

```

`\CustomAcronymFields`

```

6336 \newcommand*{\CustomAcronymFields}{%
6337 name={\the\glsshorttok},%
6338 description={\the\glslongtok},%
6339 first={\noexpand\acrfullformat{\the\glslongtok}{\the\glsshorttok}},%
6340 firstplural={\noexpand\acrfullformat
6341 {\noexpand\glsentrylongpl{\the\glslabeltok}}}%
6342 {\noexpand\glsentryshortpl{\the\glslabeltok}}},%

6343 text={\the\glsshorttok},%
6344 plural={\the\glsshorttok\noexpand\acrpluralsuffix}%
6345 }

```

`\CustomNewAcronymDef`

```

6346 \newcommand*{\CustomNewAcronymDef}{%
6347 \protected@edef\@do@newglossaryentry{%
6348 \noexpand\newglossaryentry{\the\glslabeltok}%
6349 {%

```

```

6350     type=\acronymtype,%
6351     short={\the\glsshorttok},%
6352     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6353     long={\the\glslongtok},%
6354     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6355     user1={\the\glsshorttok},%
6356     user2={\the\glsshorttok\noexpand\acrpluralsuffix},%
6357     user3={\the\glslongtok},%
6358     user4={\the\glslongtok\noexpand\acrpluralsuffix},%
6359     \CustomAcronymFields,%
6360     \the\glskeylisttok
6361 }%
6362 }%
6363 \@do@newglossaryentry
6364 }

```

\SetCustomStyle

```

6365 \newcommand*{\SetCustomStyle}{%
6366   \renewcommand{\newacronym}[4][[]]{%
6367     \ifx\@glsacronymlists\@empty
6368       \def\@glo@type{\acronymtype}%
6369       \setkeys{glossentry}{##1}%
6370       \DeclareAcronymList{\@glo@type}%
6371       \SetCustomDisplayStyle{\@glo@type}%
6372     \fi
6373     \glskeylisttok{##1}%
6374     \glslabeltok{##2}%
6375     \glsshorttok{##3}%
6376     \glslongtok{##4}%
6377     \newacronymhook
6378     \CustomNewAcronymDef
6379   }%
6380   \@for\@gls@type:=\@glsacronymlists\do{%
6381     \SetCustomDisplayStyle{\@gls@type}%
6382   }%
6383 }

```

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.

```
6384 \RequirePackage{glossary-hypernav}
```

The styles that use list-like environments. These are not loaded if the `nolist` option is used:

```
6385 \@gls@loadlist
```

The styles that use the longtable environment. These are not loaded if the no-long package option is used.

```
6386 \@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.

```
6387 \@gls@loadsuper
```

The tree-like styles. These are not loaded if the notree package option is used.

```
6388 \@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.

```
6389 \ifx\@glossary@default@style\relax
6390 \else
6391   \setglossarystyle{\@glossary@default@style}
6392 \fi
```

1.19 Debugging Commands

\showgloparent \showgloparent{<label>}

```
6393 \newcommand*\showgloparent}[1]{%
6394   \expandafter\show\csname glo@#1@parent\endcsname
6395 }
```

\showglolevel \showglolevel{<label>}

```
6396 \newcommand*\showglolevel}[1]{%
6397   \expandafter\show\csname glo@#1@level\endcsname
6398 }
```

\showglotext \showglotext{<label>}

```
6399 \newcommand*\showglotext}[1]{%
6400   \expandafter\show\csname glo@#1@text\endcsname
6401 }
```

\showgloplural \showgloplural{<label>}

```
6402 \newcommand*\showgloplural}[1]{%
6403   \expandafter\show\csname glo@#1@plural\endcsname
6404 }
```

\showglofirst \showglofirst{<label>}

```
6405 \newcommand*{\showglofirst}[1]{%
6406   \expandafter\show\csname glo@#1@first\endcsname
6407 }
```

\showglofirstpl \showglofirstpl{<label>}

```
6408 \newcommand*{\showglofirstpl}[1]{%
6409   \expandafter\show\csname glo@#1@firstpl\endcsname
6410 }
```

\showglotype \showglotype{<label>}

```
6411 \newcommand*{\showglotype}[1]{%
6412   \expandafter\show\csname glo@#1@type\endcsname
6413 }
```

\showglocounter \showglocounter{<label>}

```
6414 \newcommand*{\showglocounter}[1]{%
6415   \expandafter\show\csname glo@#1@counter\endcsname
6416 }
```

\showglouser \showglouser{<label>}

```
6417 \newcommand*{\showglouser}[1]{%
6418   \expandafter\show\csname glo@#1@useri\endcsname
6419 }
```

\showglouserii \showglouserii{<label>}

```
6420 \newcommand*{\showglouserii}[1]{%
6421   \expandafter\show\csname glo@#1@userii\endcsname
6422 }
```

\showglouseriii \showglouseriii{<label>}

```
6423 \newcommand*{\showglouseriii}[1]{%
6424   \expandafter\show\csname glo@#1@useriii\endcsname
6425 }
```

\showglouseriv \showglouseriv{<label>}

```
6426 \newcommand*{\showglouseriv}[1]{%
6427   \expandafter\show\csname glo@#1@useriv\endcsname
6428 }
```

\showglouserv \showglouserv{<label>}

```
6429 \newcommand*{\showglouserv}[1]{%
6430   \expandafter\show\csname glo@#1@userv\endcsname
6431 }
```

\showglouservi \showglouservi{<label>}

```
6432 \newcommand*{\showglouservi}[1]{%
6433   \expandafter\show\csname glo@#1@uservi\endcsname
6434 }
```

\showgloname \showgloname{<label>}

```
6435 \newcommand*{\showgloname}[1]{%
6436   \expandafter\show\csname glo@#1@name\endcsname
6437 }
```

\showglodesc \showglodesc{<label>}

```
6438 \newcommand*{\showglodesc}[1]{%
6439   \expandafter\show\csname glo@#1@desc\endcsname
6440 }
```

\showglodescplural \showglodescplural{<label>}

```
6441 \newcommand*{\showglodescplural}[1]{%
6442   \expandafter\show\csname glo@#1@descplural\endcsname
6443 }
```

\showglosort \showglosort{<label>}

```
6444 \newcommand*{\showglosort}[1]{%
6445   \expandafter\show\csname glo@#1@sort\endcsname
6446 }
```

`\showglosymbol` `\showglosymbol{<label>}`

```
6447 \newcommand*{\showglosymbol}[1]{%
6448   \expandafter\show\csname glo@#1@symbol\endcsname
6449 }
```

`\showglosymbolplural` `\showglosymbolplural{<label>}`

```
6450 \newcommand*{\showglosymbolplural}[1]{%
6451   \expandafter\show\csname glo@#1@symbolplural\endcsname
6452 }
```

`\showgloshort` `\showgloshort{<label>}`

```
6453 \newcommand*{\showgloshort}[1]{%
6454   \expandafter\show\csname glo@#1@short\endcsname
6455 }
```

`\showglolong` `\showglolong{<label>}`

```
6456 \newcommand*{\showglolong}[1]{%
6457   \expandafter\show\csname glo@#1@long\endcsname
6458 }
```

`\showgloindex` `\showgloindex{<label>}`

```
6459 \newcommand*{\showgloindex}[1]{%
6460   \expandafter\show\csname glo@#1@index\endcsname
6461 }
```

`\showgloflag` `\showgloflag{<label>}`

```
6462 \newcommand*{\showgloflag}[1]{%
6463   \expandafter\show\csname ifglo@#1@flag\endcsname
6464 }
```

`\showacronymlists` `\showacronymlists`

Show list of glossaries that have been flagged as a list of acronyms.

```

6465 \newcommand*{\showacronymlists}{%
6466   \show\@glsacronymlists
6467 }

```

\showglossaries \showglossaries

Show list of defined glossaries.

```

6468 \newcommand*{\showglossaries}{%
6469   \show\@glo@types
6470 }

```

\showglossaryin \showglossaryin{<glossary-label>}

Show the ‘in’ extension for the given glossary.

```

6471 \newcommand*{\showglossaryin}[1]{%
6472   \expandafter\show\csname @glo@type@#1@in\endcsname
6473 }

```

\showglossaryout \showglossaryout{<glossary-label>}

Show the ‘out’ extension for the given glossary.

```

6474 \newcommand*{\showglossaryout}[1]{%
6475   \expandafter\show\csname @glo@type@#1@out\endcsname
6476 }

```

\showglossarytitle \showglossarytitle{<glossary-label>}

Show the title for the given glossary.

```

6477 \newcommand*{\showglossarytitle}[1]{%
6478   \expandafter\show\csname @glo@type@#1@title\endcsname
6479 }

```

\showglossarycounter \showglossarycounter{<glossary-label>}

Show the counter for the given glossary.

```

6480 \newcommand*{\showglossarycounter}[1]{%
6481   \expandafter\show\csname @glo@type@#1@counter\endcsname
6482 }

```

\showglossaryentries \showglossaryentries{<glossary-label>}

Show the list of entry labels for the given glossary.

```
6483 \newcommand*{\showglossaryentries}[1]{%
6484   \expandafter\show\csname glolist@#1\endcsname
6485 }
```

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.

```
6486 \csname ifglscpatible-2.07\endcsname
6487   \RequirePackage{glossaries-compatible-207}
6488 \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.

```
6489 \NeedsTeXFormat{LaTeX2e}
6490 \ProvidesPackage{glossaries-prefix}[2013/11/14 v4.0 (NLCT)]
```

Pass all options to glossaries:

```
6491 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{glossaries}}
```

Process options:

```
6492 \ProcessOptions
```

Load glossaries:

```
6493 \RequirePackage{glossaries}
```

Add the new keys:

```
6494 \define@key{glossentry}{prefixfirst}{\def\@glo@entryprefixfirst{#1}}%
6495 \define@key{glossentry}{prefixfirstplural}{\def\@glo@entryprefixfirstplural{#1}}%
6496 \define@key{glossentry}{prefix}{\def\@glo@entryprefix{#1}}%
6497 \define@key{glossentry}{prefixplural}{\def\@glo@entryprefixplural{#1}}%
```

Add them to \@gls@keymap:

```
6498 \appto\@gls@keymap{,%  
6499   {prefixfirst}{prefixfirst},%  
6500   {prefixfirstplural}{prefixfirstplural},%  
6501   {prefix}{prefix},%  
6502   {prefixplural}{prefixplural}}%  
6503 }
```

Set the default values:

```
6504 \appto\@newglossaryentryprehook{%  
6505   \def\@glo@entryprefix{}%  
6506   \def\@glo@entryprefixplural{}%  
6507   \let\@glo@entryprefixfirst\@gls@default@value  
6508   \let\@glo@entryprefixfirstplural\@gls@default@value  
6509 }
```

Set the assignment code:

```
6510 \appto\@newglossaryentryposthook{%  
6511   \gls@assign@field{}\@glo@label{prefix}\@glo@entryprefix}%  
6512   \gls@assign@field{}\@glo@label{prefixplural}\@glo@entryprefixplural}%
```

If prefixfirst has not been supplied, make it the same as prefix.

```
6513   \expandafter\gls@assign@field\expandafter  
6514     {\csname glo@\@glo@label @prefix\endcsname}\@glo@label{prefixfirst}%  
6515     {\@glo@entryprefixfirst}}%
```

If prefixfirstplural has not been supplied, make it the same as prefixplural.

```
6516   \expandafter\gls@assign@field\expandafter  
6517     {\csname glo@\@glo@label @prefixplural\endcsname}\@glo@label}%  
6518     {prefixfirstplural}\@glo@entryprefixfirstplural}}%  
6519 }
```

Define commands to access these fields:

glsentryprefixfirst

```
6520 \newcommand*\glsentryprefixfirst[1]{\csuse{glo@#1@prefixfirst}}
```

glsentryprefixfirstplural

```
6521 \newcommand*\glsentryprefixfirstplural[1]{\csuse{glo@#1@prefixfirstplural}}
```

\glsentryprefix

```
6522 \newcommand*\glsentryprefix[1]{\csuse{glo@#1@prefix}}
```

glsentryprefixplural

```
6523 \newcommand*\glsentryprefixplural[1]{\csuse{glo@#1@prefixplural}}
```

Now for the initial upper case variants:

Glsentryprefixfirst

```
6524 \newrobustcmd*\Glsentryprefixfirst[1]{%  
6525   \protected@edef\@glo@text{\csname glo@#1@prefixfirst\endcsname}%  
6526   \xmakefirstuc\@glo@text  
6527 }
```

ryprefixfirstplural

```
6528 \newrobustcmd*{\Glsentryprefixfirstplural}[1]{%
6529   \protected@edef\@glo@text{\csname glo@#1@prefixfirstplural\endcsname}%
6530   \xmakefirstuc\@glo@text
6531 }
```

\Glsentryprefix

```
6532 \newrobustcmd*{\Glsentryprefix}[1]{%
6533   \protected@edef\@glo@text{\csname glo@#1@prefix\endcsname}%
6534   \xmakefirstuc\@glo@text
6535 }
```

lsentryprefixplural

```
6536 \newrobustcmd*{\Glsentryprefixplural}[1]{%
6537   \protected@edef\@glo@text{\csname glo@#1@prefixplural\endcsname}%
6538   \xmakefirstuc\@glo@text
6539 }
```

Define commands to determine if the prefix keys have been set:

\ifglshasprefix

```
6540 \newcommand*{\ifglshasprefix}[3]{%
6541   \ifcempty{glo@#1@prefix}%
6542   {#3}%
6543   {#2}%
6544 }
```

fglshasprefixplural

```
6545 \newcommand*{\ifglshasprefixplural}[3]{%
6546   \ifcempty{glo@#1@prefixplural}%
6547   {#3}%
6548   {#2}%
6549 }
```

ifglshasprefixfirst

```
6550 \newcommand*{\ifglshasprefixfirst}[3]{%
6551   \ifcempty{glo@#1@prefixfirst}%
6552   {#3}%
6553   {#2}%
6554 }
```

asprefixfirstplural

```
6555 \newcommand*{\ifglshasprefixfirstplural}[3]{%
6556   \ifcempty{glo@#1@prefixfirstplural}%
6557   {#3}%
6558   {#2}%
6559 }
```

Define commands that insert the prefix before commands like \gls:

```
\pgls
6560 \newrobustcmd{\pgls}{\@ifstar\@spgls\@pgls}
```

\@spgls Starred version.

```
6561 \newcommand*{\@spgls}[2][\@pgls@{hyper=false,#1}{#2}]
```

\@pgls Unstarred version.

```
6562 \newcommand*{\@pgls}[2][\%
6563 \new@ifnextchar[%
6564 {\@pgls@{#1}{#2}}%
6565 {\@pgls@{#1}{#2}[]}%
6566 }
```

\@pgls@ Read in the final optional argument:

```
6567 \def\@pgls@#1#2[#3]{%
6568 \glsdoifexists{#2}%
6569 {%
6570 \ifglsused{#2}%
6571 {%
6572 \glsentryprefix{#2}%
6573 }%
6574 {%
6575 \glsentryprefixfirst{#2}%
6576 }%
6577 \@gls@{#1}{#2}[#3]%
6578 }%
6579 }
```

Similarly for the plural version:

```
\pglsp1
6580 \newrobustcmd{\pglsp1}{\@ifstar\@spglsp1\@pglsp1}
```

\@spglsp1 Starred version.

```
6581 \newcommand*{\@spglsp1}[2][\@pglsp1@{hyper=false,#1}{#2}]
```

\@pglsp1 Unstarred version.

```
6582 \newcommand*{\@pglsp1}[2][\%
6583 \new@ifnextchar[%
6584 {\@pglsp1@{#1}{#2}}%
6585 {\@pglsp1@{#1}{#2}[]}%
6586 }
```

\@pglsp1@ Read in the final optional argument:

```
6587 \def\@pglsp1@#1#2[#3]{%
6588 \glsdoifexists{#2}%
6589 {%
6590 \ifglsused{#2}%
```

```

6591    {%
6592        \glsentryprefixplural{#2}%
6593    }%
6594    {%
6595        \glsentryprefixfirstplural{#2}%
6596    }%
6597    \@glspl@{#1}{#2}[#3]%
6598 }%
6599 }

```

Now for the first letter upper case versions:

\Pgl's

```

6600 \newrobustcmd{\Pgl's}{\@ifstar\@sPgl's\@Pgl's}

```

\@sPgl's Starred version.

```

6601 \newcommand*{\@sPgl's}[2][\@Pgl's@{hyper=false,#1}{#2}]

```

\@Pgl's Unstarred version.

```

6602 \newcommand*{\@Pgl's}[2][\%
6603     \new@ifnextchar[\%
6604     {\@Pgl's@{#1}{#2}}\%
6605     {\@Pgl's@{#1}{#2}}[]\%
6606 ]

```

\@Pgl's@ Read in the final optional argument:

```

6607 \def\@Pgl's@#1#2[#3]{\%
6608     \glsdoifexists{#2}%
6609     {\%
6610         \ifgl'sused{#2}%
6611         {\%
6612             \ifgl'shasprefix{#2}%
6613             {\%
6614                 \Glsentryprefix{#2}%
6615                 \@gls@{#1}{#2}[#3]%
6616             }%
6617             {\@Gls@{#1}{#2}[#3]}%
6618         }%
6619         {\%
6620             \ifgl'shasprefixfirst{#2}%
6621             {\%
6622                 \Glsentryprefixfirst{#2}%
6623                 \@gls@{#1}{#2}[#3]%
6624             }%
6625             {\@Gls@{#1}{#2}[#3]}%
6626         }%
6627     }%
6628 }

```

Similarly for the plural version:

```
\Pglsp1
6629 \newrobustcmd{\Pglsp1}{\@ifstar\@sPglsp1\@Pglsp1}

\@sPglsp1  Starred version.
6630 \newcommand*{\@sPglsp1}[2][\@Pglsp1@{hyper=false,#1}{#2}]

\@Pglsp1  Unstarred version.
6631 \newcommand*{\@Pglsp1}[2][\%
6632   \new@ifnextchar[\%
6633   {\@Pglsp1@{#1}{#2}}\%
6634   {\@Pglsp1@{#1}{#2}[]}\%
6635 ]

\@Pglsp1@  Read in the final optional argument:
6636 \def\@Pglsp1@#1#2[#3]{\%
6637   \glsdoifexists{#2}\%
6638   {\%
6639     \ifglsused{#2}\%
6640     {\%
6641       \ifgls hasprefixplural{#2}\%
6642       {\%
6643         \Glsentryprefixplural{#2}\%
6644         \@glspl@{#1}{#2}[#3]\%
6645       }\%
6646       {\@Glspl@{#1}{#2}[#3]}\%
6647     }\%
6648     {\%
6649       \ifgls hasprefixfirstplural{#2}\%
6650       {\%
6651         \Glsentryprefixfirstplural{#2}\%
6652         \@glspl@{#1}{#2}[#3]\%
6653       }\%
6654       {\@Glspl@{#1}{#2}[#3]}\%
6655     }\%
6656   }\%
6657 }
```

Finally the all upper case versions:

```
\PGLS
6658 \newrobustcmd{\PGLS}{\@ifstar\@sPGLS\@PGLS}

\@sPGLS  Starred version.
6659 \newcommand*{\@sPGLS}[2][\@PGLS@{hyper=false,#1}{#2}]
```

\@PGLS Unstarred version.

```
6660 \newcommand*{\@PGLS}[2][\%  
6661   \new@ifnextchar[\  
6662   {\@PGLS@{#1}{#2}}%  
6663   {\@PGLS@{#1}{#2}[]}%  
6664 }
```

\@PGLS@ Read in the final optional argument:

```
6665 \def\@PGLS@#1#2[#3]{%  
6666   \glsdoifexists{#2}%  
6667   {%  
6668     \ifglsused{#2}%  
6669     {%  
6670       \mfirstucMakeUppercase{\glsentryprefix{#2}}%  
6671     }%  
6672     {%  
6673       \mfirstucMakeUppercase{\glsentryprefixfirst{#2}}%  
6674     }%  
6675     \@GLS@{#1}{#2}[#3]%  
6676   }%  
6677 }
```

Plural version:

\PGLSp1

```
6678 \newrobustcmd{\PGLSp1}{\@ifstar\@sPGLSp1\@PGLSp1}
```

\@sPGLSp1 Starred version.

```
6679 \newcommand*{\@sPGLSp1}[2][\@PGLSp1@{hyper=false,#1}{#2}]
```

\@PGLSp1 Unstarred version.

```
6680 \newcommand*{\@PGLSp1}[2][\%  
6681   \new@ifnextchar[\  
6682   {\@PGLSp1@{#1}{#2}}%  
6683   {\@PGLSp1@{#1}{#2}[]}%  
6684 }
```

\@PGLSp1@ Read in the final optional argument:

```
6685 \def\@PGLSp1@#1#2[#3]{%  
6686   \glsdoifexists{#2}%  
6687   {%  
6688     \ifglsused{#2}%  
6689     {%  
6690       \mfirstucMakeUppercase{\glsentryprefixplural{#2}}%  
6691     }%  
6692     {%  
6693       \mfirstucMakeUppercase{\glsentryprefixfirstplural{#2}}%  
6694     }%  
6695     \@GLSp1@{#1}{#2}[#3]%  
6696   }%  
6697 }
```

```
6696 }%
6697 }
```

3 Mfirstuc Documented Code

```
6698 \NeedsTeXFormat{LaTeX2e}
6699 \ProvidesPackage{mfirstuc}[2013/11/04 v1.08 (NLCT)]
```

Requires etoolbox:

```
6700 \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.

```
6701 \newif\if@glscs
6702 \newtoks\@glsmfirst
6703 \newtoks\@glsmrest
6704 \newrobustcmd*{\makefirstuc}[1]{%
6705   \def\gls@argi{#1}%
6706   \ifx\gls@argi\@empty

   If the argument is empty, do nothing.
6707   \else

6708   \def\@gls@tmp{\ #1}%
6709   \@onelevel@sanitize\@gls@tmp
6710   \expandafter\@gls@checkcs\@gls@tmp\relax\relax
6711   \if@glscs
6712     \@gls@getbody #1{}\@nil
6713     \ifx\@gls@rest\@empty
6714       \glsmakefirstuc{#1}%
6715     \else
6716       \expandafter\@gls@split\@gls@rest\@nil
6717       \ifx\@gls@first\@empty
6718         \glsmakefirstuc{#1}%
6719       \else
6720         \expandafter\@glsmfirst\expandafter{\@gls@first}%
6721         \expandafter\@glsmrest\expandafter{\@gls@rest}%
6722         \edef\@gls@domfirstuc{\noexpand\@gls@body
6723           {\noexpand\glsmakefirstuc\the\@glsmfirst}%
6724           \the\@glsmrest}%
6725         \@gls@domfirstuc
6726       \fi
```

```

6727     \fi
6728     \else
6729     \glsmakefirstuc{#1}%
6730     \fi
6731 \fi
6732 }

```

Put first argument in \@gls@first and second argument in \@gls@rest:

```

6733 \def\@gls@split#1#2\@nil{%
6734   \def\@gls@first{#1}\def\@gls@rest{#2}%
6735 }

6736 \def\@gls@checkcs#1 #2#3\relax{%
6737   \def\@gls@argi{#1}\def\@gls@argii{#2}%
6738   \ifx\@gls@argi\@gls@argii
6739     \@glscstrue
6740   \else
6741     \@glscsfalse
6742   \fi
6743 }

```

\@gls@makefirstuc Make first thing upper case:

```

6744 \def\@gls@makefirstuc#1{\mfirstucMakeUppercase #1}

```

mfirstucMakeUppercase Allow user to replace \MakeUppercase with another case changing command.

```

6745 \newcommand*{\mfirstucMakeUppercase}{\MakeUppercase}

```

\glsmakefirstuc Provide a user command to make it easier to customise.

```

6746 \newcommand*{\glsmakefirstuc}[1]{\@gls@makefirstuc{#1}}

```

Get the first grouped argument and stores in \@gls@body.

```

6747 \def\@gls@getbody#1#{\def\@gls@body{#1}\@gls@gobbletonil}

```

Scoup up everything to \@nil and store in \@gls@rest:

```

6748 \def\@gls@gobbletonil#1\@nil{\def\@gls@rest{#1}}

```

\xmakefirstuc Expand argument once before applying \makefirstuc (added v1.01).

```

6749 \newcommand*{\xmakefirstuc}[1]{%
6750 \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).

```

6751 \newrobustcmd*{\capitalisewords}[1]{%
6752   \def\gls@add@space{}%
6753   \mfu@capitalisewords#1 \@nil\mfu@endcap
6754 }

```

```

6755 \def\mfu@capitalisewords#1 #2\mfu@endcap{%
6756   \def\mfu@cap@first{#1}%
6757   \def\mfu@cap@second{#2}%
6758   \gls@add@space
6759   \makefirstuc{#1}%
6760   \def\gls@add@space{ }%
6761   \ifx\mfu@cap@second\@nnil
6762     \let\next\mfu@cap\mfu@noop
6763   \else
6764     \let\next\mfu@cap\mfu@capitalisewords
6765   \fi
6766   \next\mfu@cap#2\mfu@endcap
6767 }
6768 \def\mfu@noop#1\mfu@endcap{}

```

\xcapitalisewords Short-cut command:

```

6769 \newcommand*{\xcapitalisewords}[1]{%
6770   \expandafter\capitalisewords\expandafter{#1}%
6771 }

```

4 Glossary Styles

4.1 Glossary hyper-navigation definitions (glossary-hypernav package)

Package Definition:

```

6772 \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

```

6773 \newcommand*{\glsnavhyperlink}[3][\@glo@type]{%
6774   \edef\gls@grplabel{#2}\protected@edef\gls@grptitle{#3}%
6775   \@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
6776 \newcommand*{\glsnavhypertarget}[3][\@glo@type]{%
  Add this group to the aux file for re-run check.
6777   \protected@write\@auxout{}\string\@gls@hypergroup{#1}{#2}}%
  Add the target.
6778   \@glstarget{glsn:#1@#2}{#3}%
  Check list of know groups to determine if a re-run is required.
6779   \expandafter\let
6780     \expandafter\@gls@list\csname @gls@hypergroup@list@#1\endcsname
  Iterate through list and terminate loop if this group is found.
6781   \@for\@gls@elem:=\@gls@list\do{%
6782     \ifthenelse{\equal{\@gls@elem}{#2}}{\@endfortrue}{}}%
  Check if list terminated prematurely.
6783   \if@endfor
6784   \else
    This group was not included in the list, so issue a warning.
6785     \GlossariesWarningNoLine{Navigation panel
6786       for glossary type ‘#1’^^Jmissing group ‘#2’}%
6787     \gdef\gls@hypergroup@rerun{%
6788       \GlossariesWarningNoLine{Navigation panel
6789         has changed. Rerun LaTeX}}%
6790   \fi
6791 }

\gls@hypergroup@rerun Give a warning at the end if re-run required
6792 \let\gls@hypergroup@rerun\relax
6793 \AtEndDocument{\gls@hypergroup@rerun}

\@gls@hypergroup This adds to (or creates) the command \@gls@hypergroup@list@{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.
6794 \newcommand*{\@gls@hypergroup}[2]{%
6795 \ifundefined{\@gls@hypergroup@list@#1}{%
6796   \expandafter\xdef\csname @gls@hypergroup@list@#1\endcsname{#2}%
6797 }{%
6798   \expandafter\let\expandafter\@gls@tmp
6799     \csname @gls@hypergroup@list@#1\endcsname
6800   \expandafter\xdef\csname @gls@hypergroup@list@#1\endcsname{%
6801     \@gls@tmp,#2}%
6802 }%
6803 }

```

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`

```
6804 \newcommand*\glsnavigation{%
6805 \def\@gls@between{%
6806 \@ifundefined{\@gls@hypergroup@list\@glo@type}{%
6807   \def\@gls@list{%
6808 }{%
6809   \expandafter\let\expandafter\@gls@list
6810     \csname \@gls@hypergroup@list\@glo@type\endcsname
6811 }%
6812 \@for\@gls@tmp:=\@gls@list\do{%
6813   \@gls@between

6814   \@gls@getgrouptitle{\@gls@tmp}{\@gls@grptitle}%
6815   \glsnavhyperlink{\@gls@tmp}{\@gls@grptitle}%
6816   \let\@gls@between\glshypernavsep%
6817 }%
6818 }
```

`\glshypernavsep` Separator for the hyper navigation bar.

```
6819 \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`

```
6820 \newcommand*\glssymbolnav{%
6821 \glsnavhyperlink{glssymbols}{\glsgetgrouptitle{glssymbols}}%
6822 \glshypernavsep
6823 \glsnavhyperlink{glsnumbers}{\glsgetgrouptitle{glsnumbers}}%
6824 \glshypernavsep
6825 }
```

4.2 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.

```
6826 \ProvidesPackage{glossary-inline}[2013/11/14 v4.0 (NLCT)]
```

`inline` Define the inline style.

```
6827 \newglossarystyle{inline}{%
```

Start of glossary sets up first empty separator between entries. (This is then changed by `\glossentry`)

```
6828 \renewenvironment{theglossary}%
6829 {%
6830     \def\gls@inlinesep{}%
6831     \def\gls@inlinesubsep{}%
6832     \def\gls@inlinepostchild{}%
6833 }%
6834 {\glspostinline}%
```

No header:

```
6835 \renewcommand*{\glossaryheader}{}%
```

No group headings (if heading is required, add `\glsinlinedopostchild` to start definition in case heading follows a child entry):

```
6836 \renewcommand*{\glsgroupheading}[1]{}%
```

Just display separator followed by name and description:

```
6837 \renewcommand{\glossentry}[2]{%
6838     \glsinlinedopostchild
6839     \gls@inlinesep
6840     \glsentryitem{##1}%
6841     \glsinlinenameformat{##1}{%
6842         \glossentryname{##1}%
6843     }%
6844     \ifglshasdescsuppressed{##1}%
6845     {%
6846         \glsinlineemptydescformat
6847         {%
6848             \glossentrysymbol{##1}%
6849         }%
6850         {%
6851             ##2%
6852         }%
6853     }%
6854     {%
6855         \ifglshasdesc{##1}%
6856         {\glsinlinedescformat{\glossentrydesc{##1}}{\glossentrysymbol{##1}}{##2}}%
6857         {\glsinlineemptydescformat{\glossentrysymbol{##1}}{##2}}%
6858     }%
6859     \ifglshaschildren{##1}%
6860     {%
6861         \glsresetsubentrycounter
6862         \glsinlineparentchildseparator
6863         \def\gls@inlinesubsep{}%
6864         \def\gls@inlinepostchild{\glsinlinepostchild}%
6865     }%
6866     {}%
6867     \def\gls@inlinesep{\glsinlineseparator}%
6868 }%
```

Sub-entries display description:

```
6869 \renewcommand{\subglossentry}[3]{%
6870   \gls@inlinesubsep%
6871   \glsinlinesubnameformat{##2}{%
6872     \glossentryname{##2}}%
6873   \glssubentryitem{##2}%
6874   \glsinlinesubdescformat{\glossentrydesc{##2}}{\glossentrysymbol{##2}}{##3}%
6875   \def\gls@inlinesubsep{\glsinlinesubseparator}%
6876 }%
```

Nothing special between groups:

```
6877 \renewcommand*{\glsgroupskip}{}%
6878 }
```

`\glsinlinedopostchild`

```
6879 \newcommand*{\glsinlinedopostchild}{%
6880   \gls@inlinepostchild
6881   \def\gls@inlinepostchild{}%
6882 }
```

`\glsinlineseparator` Separator to use between entries.

```
6883 \newcommand*{\glsinlineseparator}{;\space}
```

`\glsinlinesubseparator` Separator to use between sub-entries.

```
6884 \newcommand*{\glsinlinesubseparator}{,\space}
```

`\glsinlinparentchildseparator` Separator to use between parent and children.

```
6885 \newcommand*{\glsinlinparentchildseparator}{:\space}
```

`\glsinlinepostchild` Hook to use between child and next entry

```
6886 \newcommand*{\glsinlinepostchild}{}
```

`\glsinlinepost` Terminator for inline glossary.

```
6887 \newcommand*{\glsinlinepost}{\glsinlinepostdescription\space}
```

`\glsinlinenameformat` Formats the name of the entry (first argument label, second argument name):

```
6888 \newcommand*{\glsinlinenameformat}[2]{\glstarget{#1}{#2}}
```

`\glsinlinedescformat` Formats the entry's description, symbol and location list:

```
6889 \newcommand*{\glsinlinedescformat}[3]{\space#1}
```

`\glsinlineemptydescformat` Formats the entry's symbol and location list when the description is empty:

```
6890 \newcommand*{\glsinlineemptydescformat}[2]{}
```

`\glsinlinesubnameformat` Formats the name of the subentry (first argument label, second argument name):

```
6891 \newcommand*{\glsinlinesubnameformat}[2]{\glstarget{#1}{}}
```

`\glsinlinesubdescformat` Formats the subentry's description, symbol and location list:

```
6892 \newcommand*{\glsinlinesubdescformat}[3]{#1}
```

4.3 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.

```
6893 \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.

```
6894 \newglossarystyle{list}{%
```

Use description environment:

```
6895 \renewenvironment{theglossary}{%
```

```
6896 \begin{description}}{\end{description}}%
```

No header at the start of the environment:

```
6897 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
6898 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries start a new item in the list:

```
6899 \renewcommand*{\glossentry}[2]{%
```

```
6900 \item[\glsentryitem{##1}]%
```

```
6901 \glstarget{##1}{\glossentryname{##1}}]
```

```
6902 \glossentrydesc{##1}\glspostdescription\space ##2}%
```

Sub-entries continue on the same line:

```
6903 \renewcommand*{\subglossentry}[3]{%
```

```
6904 \glssubentryitem{##2}]%
```

```
6905 \glstarget{##2}{\strut}]%
```

```
6906 \glossentrydesc{##2}\glspostdescription\space ##3.}%
```

```
6907% \end{macrocode}
```

```
6908% Add vertical space between groups:
```

```
6909%\changes{3.03}{2012/09/21}{added check for glsnogroupskip}
```

```
6910% \begin{macrocode}
```

```
6911 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
```

```
6912}
```

listgroup The listgroup style is like the list style, but the glossary groups have headings.

```
6913 \newglossarystyle{listgroup}{%
```

Base it on the list style:

```
6914 \setglossarystyle{list}%
```

Each group has a heading:

```
6915 \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.

```
6916 \newglossarystyle{listhypergroup}{%
```

Base it on the list style:

```
6917 \setglossarystyle{list}{%
```

Add navigation links at the start of the environment:

```
6918 \renewcommand*{\glossaryheader}{%
```

```
6919 \item[\glsnavigation]}%
```

Each group has a heading with a hypertarget:

```
6920 \renewcommand*{\glsgroupheading}[1]{%
```

```
6921 \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.

```
6922 \newglossarystyle{altlist}{%
```

Base it on the list style:

```
6923 \setglossarystyle{list}{%
```

Main (level 0) entries start a new item in the list with a line break after the entry name:

```
6924 \renewcommand*{\glossentry}[2]{%
```

```
6925 \item[\glsentryitem{##1}%
```

```
6926 \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.

```
6927 \mbox{}\par\nobreak\@afterheading
```

```
6928 \glossentrydesc{##1}\glspostdescription\space ##2}%
```

Sub-entries start a new paragraph:

```
6929 \renewcommand{\subglossentry}[3]{%
```

```
6930 \par
```

```
6931 \glssubentryitem{##2}%
```

```
6932 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space ##3}%
```

```
6933 }
```

altlistgroup The altlistgroup glossary style is like the altlist style, but the glossary groups have headings.

```
6934 \newglossarystyle{altlistgroup}{%
```

Base it on the altlist style:

```
6935 \setglossarystyle{altlist}{%
```

Each group has a heading:

```
6936 \renewcommand*{\glsgroupheading}[1]{\item[\glsgetgrouptitle{##1}]}}
```

`altlisthypergroup` The `altlisthypergroup` glossary style is like the `altlistgroup` style, but has a set of links to the groups at the start of the glossary.

```
6937 \newglossarystyle{altlisthypergroup}{%
```

Base it on the `altlist` style:

```
6938 \setglossarystyle{altlist}%
```

Add navigation links at the start of the environment:

```
6939 \renewcommand*{\glossaryheader}{%
```

```
6940 \item[\glsnavigation]}%
```

Each group has a heading with a `hypertarget`:

```
6941 \renewcommand*{\glsgroupheading}[1]{%
```

```
6942 \item[\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}]}
```

`listdotted` The `listdotted` glossary style was supplied by Axel Menzel. I've modified it slightly so that the distance from the start of the name to the end of the dotted line is specified by `\glslistdottedwidth`. Note that this style ignores the page numbers as well as the symbol. Sub-entries are displayed in the same way as top-level entries.

```
6943 \newglossarystyle{listdotted}{%
```

Base it on the `list` style:

```
6944 \setglossarystyle{list}%
```

Each main (level 0) entry starts a new item:

```
6945 \renewcommand*{\glossentry}[2]{%
```

```
6946 \item[\makebox[\glslistdottedwidth][l]{%
```

```
6947 \glsentryitem{##1}%
```

```
6948 \glstarget{##1}{\glossentryname{##1}}%
```

```
6949 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}\glossentrydesc{##1}}%
```

Sub entries have the same format as main entries:

```
6950 \renewcommand*{\subglossentry}[3]{%
```

```
6951 \item[\makebox[\glslistdottedwidth][l]{%
```

```
6952 \glssubentryitem{##2}%
```

```
6953 \glstarget{##2}{\glossentryname{##2}}%
```

```
6954 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}\glossentrydesc{##2}}%
```

```
6955 }
```

`\glslistdottedwidth`

```
6956 \newlength\glslistdottedwidth
```

```
6957 \setlength{\glslistdottedwidth}{.5\hsize}
```

`sublistdotted` This style is similar to the `glostylelistdotted` style, except that the main entries just have the name displayed.

```
6958 \newglossarystyle{sublistdotted}{%
```

Base it on the `listdotted` style:

```
6959 \setglossarystyle{listdotted}%
```

Main (level 0) entries just display the name:

```
6960 \renewcommand*{\glossentry}[2]{%
6961   \item[\glentryitem{##1}\glstarget{##1}{\glossentryname{##1}}]}%
6962 }
```

4.4 Glossary Styles using longtable (the glossary-long package)

The glossary styles defined in the package used the longtable environment in the glossary.

```
6963 \ProvidesPackage{glossary-long}[2013/11/14 v4.0 (NLCT)]
```

Requires the package:

```
6964 \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`.)

```
6965 \@ifundefined{glsdescwidth}{%
6966   \newlength\glsdescwidth
6967   \setlength{\glsdescwidth}{0.6\hsize}
6968 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column.

```
6969 \@ifundefined{glspagelistwidth}{%
6970   \newlength\glspagelistwidth
6971   \setlength{\glspagelistwidth}{0.1\hsize}
6972 }{}
```

`long` The long glossary style command which uses the longtable environment:

```
6973 \newglossarystyle{long}{%
```

Use longtable with two columns:

```
6974   \renewenvironment{theglossary}%
6975     {\begin{longtable}\lp{\glsdescwidth}}%
6976     {\end{longtable}}%
```

Do nothing at the start of the environment:

```
6977   \renewcommand*{\glossaryheader}{}%
```

No heading between groups:

```
6978   \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries displayed in a row:

```
6979   \renewcommand{\glossentry}[2]{%
6980     \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
6981     \glossentrydesc{##1}\glspostdescription\space ##2\tabularnewline
6982   }%
```

Sub entries displayed on the following row without the name:

```
6983 \renewcommand{\subglossentry}[3]{%
6984     &
6985     \glssubentryitem{##2}%
6986     \glstarget{##2}{\strut}\glosentrydesc{##2}\glspostdescription\space
6987     ##3\tabularnewline
6988 }
```

Blank row between groups:

```
6989 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else &
6990 \tabularnewline\fi}%
6991 }
```

longborder The longborder style is like the above, but with horizontal and vertical lines:

```
6992 \newglossarystyle{longborder}{%
    Base it on the glostylelong style:
6993 \setglossarystyle{long}%
    Use longtable with two columns with vertical lines between each column:
6994 \renewenvironment{theglossary}{%
6995     \begin{longtable}{|l|p{\glsdescwidth}|}{\end{longtable}}%
    Place horizontal lines at the head and foot of the table:
6996 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
6997 }
```

longheader The longheader style is like the long style but with a header:

```
6998 \newglossarystyle{longheader}{%
    Base it on the glostylelong style:
6999 \setglossarystyle{long}%
    Set the table's header:
7000 \renewcommand*{\glossaryheader}{%
7001     \bfseries \entryname & \bfseries \descriptionname\tabularnewline\endhead}%
7002 }
```

longheaderborder The longheaderborder style is like the long style but with a header and border:

```
7003 \newglossarystyle{longheaderborder}{%
    Base it on the glostylelongborder style:
7004 \setglossarystyle{longborder}%
    Set the table's header and add horizontal line to table's foot:
7005 \renewcommand*{\glossaryheader}{%
7006     \hline\bfseries \entryname & \bfseries
7007     \descriptionname\tabularnewline\hline
7008     \endhead
7009     \hline\endfoot}%
7010 }
```

long3col The long3col style is like long but with 3 columns

```
7011 \newglossarystyle{long3col}{%  
    Use a longtable with 3 columns:  
7012     \renewenvironment{theglossary}%  
7013         {\begin{longtable}{lp{\glstdescwidth}p{\glspagelistwidth}}}%  
7014         {\end{longtable}}}%
```

No table header:

```
7015 \renewcommand*{\glossaryheader}{}%
```

No headings between groups:

```
7016 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
7017 \renewcommand{\glossentry}[2]{%  
7018     \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &  
7019     \glossentrydesc{##1} & ##2\tabularnewline  
7020 }%
```

Sub-entries on a separate row (no name, description in second column, page list in third column):

```
7021 \renewcommand{\subglossentry}[3]{%  
7022     &  
7023     \glssubentryitem{##2}%  
7024     \glstarget{##2}{\strut}\glossentrydesc{##2} &  
7025     ##3\tabularnewline  
7026 }%
```

Blank row between groups:

```
7027 \renewcommand*{\glsgroupskip}{%  
7028     \ifglsnogroupskip\else & &\tabularnewline\fi}%  
7029 }
```

long3colborder The long3colborder style is like the long3col style but with a border:

```
7030 \newglossarystyle{long3colborder}{%  
    Base it on the glostylelong3col style:  
7031     \setglossarystyle{long3col}%  
    Use a longtable with 3 columns with vertical lines around them:  
7032     \renewenvironment{theglossary}%  
7033         {\begin{longtable}{|lp{\glstdescwidth}|p{\glspagelistwidth}|}}%  
7034         {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
7035 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%  
7036 }
```

long3colheader The long3colheader style is like long3col but with a header row:

```
7037 \newglossarystyle{long3colheader}{%
```

Base it on the `glostylelong3col` style:

```
7038 \setglossarystyle{long3col}%
```

Set the table's header:

```
7039 \renewcommand*{\glossaryheader}{%  
7040 \bfseries\entryname&\bfseries\descriptionname&  
7041 \bfseries\pagelistname\tabularnewline\endhead}%  
7042 }
```

`long3colheaderborder` The `long3colheaderborder` style is like the above but with a border

```
7043 \newglossarystyle{long3colheaderborder}{%
```

Base it on the `glostylelong3colborder` style:

```
7044 \setglossarystyle{long3colborder}%
```

Set the table's header and add horizontal line at table's foot:

```
7045 \renewcommand*{\glossaryheader}{%  
7046 \hline  
7047 \bfseries\entryname&\bfseries\descriptionname&  
7048 \bfseries\pagelistname\tabularnewline\hline\endhead  
7049 \hline\endfoot}%  
7050 }
```

`long4col` The `long4col` style has four columns where the third column contains the value of the associated symbol key.

```
7051 \newglossarystyle{long4col}{%
```

Use a longtable with 4 columns:

```
7052 \renewenvironment{theglossary}%  
7053 {\begin{longtable}{llll}}%  
7054 {\end{longtable}}%
```

No table header:

```
7055 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7056 \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):

```
7057 \renewcommand{\glossentry}[2]{%  
7058 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &  
7059 \glossentrydesc{##1} &  
7060 \glossentrysymbol{##1} &  
7061 ##2\tabularnewline  
7062 }%
```

Sub entries on a single row with no name (description in second column, symbol in third column, page list in last column):

```
7063 \renewcommand{\subglossentry}[3]{%  
7064 &
```

```

7065     \glssubentryitem{##2}%
7066     \glstarget{##2}{\strut}\glossentrydesc{##2} &
7067     \glossentrysymbol{##2} & ##3\tabularnewline
7068 }%

```

Blank row between groups:

```

7069 \renewcommand*{\glsgroupskip}{%
7070     \ifglsgnogroupskip\else & & \tabularnewline\fi}%
7071 }

```

long4colheader The long4colheader style is like long4col but with a header row.

```

7072 \newglossarystyle{long4colheader}{%

```

Base it on the glostylelong4col style:

```

7073 \setglossarystyle{long4col}%

```

Table has a header:

```

7074 \renewcommand*{\glossaryheader}{%
7075     \bfseries\entryname&\bfseries\descriptionname&
7076     \bfseries \symbolname&
7077     \bfseries\pagelistname\tabularnewline\endhead}%
7078 }

```

long4colborder The long4colborder style is like long4col but with a border.

```

7079 \newglossarystyle{long4colborder}{%

```

Base it on the glostylelong4col style:

```

7080 \setglossarystyle{long4col}%

```

Use a longtable with 4 columns surrounded by vertical lines:

```

7081 \renewenvironment{theglossary}%
7082     {\begin{longtable}{|l|l|l|l|}}%
7083     {\end{longtable}}%

```

Add horizontal lines to the head and foot of the table:

```

7084 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
7085 }

```

long4colheaderborder The long4colheaderborder style is like the above but with a border.

```

7086 \newglossarystyle{long4colheaderborder}{%

```

Base it on the glostylelong4col style:

```

7087 \setglossarystyle{long4col}%

```

Use a longtable with 4 columns surrounded by vertical lines:

```

7088 \renewenvironment{theglossary}%
7089     {\begin{longtable}{|l|l|l|l|}}%
7090     {\end{longtable}}%

```

Add table header and horizontal line at the table's foot:

```

7091 \renewcommand*{\glossaryheader}{%
7092     \hline\bfseries\entryname&\bfseries\descriptionname&

```

```

7093 \bfseries \symbolname&
7094 \bfseries\pagelistname\tabularnewline\hline\endhead
7095 \hline\endfoot}%
7096 }

```

altlong4col The altlong4col style is like the long4col style but can have multiline descriptions and page lists.

```

7097 \newglossarystyle{altlong4col}{%
    Base it on the glostylelong4col style:
7098 \setglossarystyle{long4col}%
    Use a longtable with 4 columns where the second and last columns may have
    multiple lines in each row:
7099 \renewenvironment{theglossary}%
7100 {\begin{longtable}{lp{\glsgdescwidth}lp{\glspagelistwidth}}}%
7101 {\end{longtable}}%
7102 }

```

altlong4colheader The altlong4colheader style is like altlong4col but with a header row.

```

7103 \newglossarystyle{altlong4colheader}{%
    Base it on the glostylelong4colheader style:
7104 \setglossarystyle{long4colheader}%
    Use a longtable with 4 columns where the second and last columns may have
    multiple lines in each row:
7105 \renewenvironment{theglossary}%
7106 {\begin{longtable}{lp{\glsgdescwidth}lp{\glspagelistwidth}}}%
7107 {\end{longtable}}%
7108 }

```

altlong4colborder The altlong4colborder style is like altlong4col but with a border.

```

7109 \newglossarystyle{altlong4colborder}{%
    Base it on the glostylelong4colborder style:
7110 \setglossarystyle{long4colborder}%
    Use a longtable with 4 columns where the second and last columns may have
    multiple lines in each row:
7111 \renewenvironment{theglossary}%
7112 {\begin{longtable}{|lp{\glsgdescwidth}|lp{\glspagelistwidth}|}}%
7113 {\end{longtable}}%
7114 }

```

altlong4colheaderborder The altlong4colheaderborder style is like the above but with a header as well as a border.

```

7115 \newglossarystyle{altlong4colheaderborder}{%
    Base it on the glostylelong4colheaderborder style:
7116 \setglossarystyle{long4colheaderborder}%

```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```

7117 \renewenvironment{theglossary}%
7118   {\begin{longtable}{|l|p{\glsdescwidth}|l|p{\glspagelistwidth}|}%
7119   {\end{longtable}}%
7120 }
```

4.5 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.

```

7121 \ProvidesPackage{glossary-longragged}[2013/11/14 v4.0 (NLCT)]
```

Requires the package:

```

7122 \RequirePackage{array}
```

Requires the package:

```

7123 \RequirePackage{longtable}
```

`\glsdescwidth` This is a length that governs the width of the description column. This may have already been defined.

```

7124 \@ifundefined{glsdescwidth}{%
7125   \newlength\glsdescwidth
7126   \setlength{\glsdescwidth}{0.6\hsize}
7127 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

```

7128 \@ifundefined{glspagelistwidth}{%
7129   \newlength\glspagelistwidth
7130   \setlength{\glspagelistwidth}{0.1\hsize}
7131 }{}
```

`longragged` The longragged glossary style is like the long but uses ragged right formatting for the description column.

```

7132 \newglossarystyle{longragged}{%
```

Use longtable with two columns:

```

7133 \renewenvironment{theglossary}%
7134   {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}}}%
7135   {\end{longtable}}%
```

Do nothing at the start of the environment:

```

7136 \renewcommand*{\glossaryheader}{}%
```

No heading between groups:

```

7137 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries displayed in a row:

```

7138 \renewcommand{\glossentry}[2]{%
7139     \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7140     \glossentrydesc{##1}\glspostdescription\space ##2%
7141     \tabularnewline
7142 }%
```

Sub entries displayed on the following row without the name:

```

7143 \renewcommand{\subglossentry}[3]{%
7144     &
7145     \glssubentryitem{##2}%
7146     \glstarget{##2}{\strut}\glossentrydesc{##2}%
7147     \glspostdescription\space ##3%
7148     \tabularnewline
7149 }%
```

Blank row between groups:

```

7150 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & \tabularnewline\fi}%
7151 }
```

longraggedborder The longraggedborder style is like the above, but with horizontal and vertical lines:

```

7152 \newglossarystyle{longraggedborder}{%
```

Base it on the glostylelongragged style:

```

7153 \setglossarystyle{longragged}%
```

Use longtable with two columns with vertical lines between each column:

```

7154 \renewenvironment{theglossary}{%
7155     \begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|}%
7156     {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```

7157 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
7158 }
```

longraggedheader The longraggedheader style is like the longragged style but with a header:

```

7159 \newglossarystyle{longraggedheader}{%
```

Base it on the glostylelongragged style:

```

7160 \setglossarystyle{longragged}%
```

Set the table's header:

```

7161 \renewcommand*{\glossaryheader}{%
7162     \bfseries \entryname & \bfseries \descriptionname
7163     \tabularnewline\endhead}%
7164 }
```

raggedheaderborder The longraggedheaderborder style is like the longragged style but with a header and border:

```

7165 \newglossarystyle{longraggedheaderborder}{%
```

Base it on the `glostylelongraggedborder` style:

```
7166 \setglossarystyle{longraggedborder}%
```

Set the table's header and add horizontal line to table's foot:

```
7167 \renewcommand*{\glossaryheader}{%
7168   \hline\bfseries \entryname & \bfseries \descriptionname
7169   \tabularnewline\hline
7170   \endhead
7171   \hline\endfoot}%
7172 }
```

`longragged3col` The `longragged3col` style is like `longragged` but with 3 columns

```
7173 \newglossarystyle{longragged3col}{%
```

Use a longtable with 3 columns:

```
7174 \renewenvironment{theglossary}%
7175   {\begin{longtable}{l>{\raggedright}p{\glsgdescwidth}%
7176     >{\raggedright}p{\glspagelistwidth}}}%
7177   {\end{longtable}}%
```

No table header:

```
7178 \renewcommand*{\glossaryheader}{}%
```

No headings between groups:

```
7179 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
7180 \renewcommand{\glossentry}[2]{%
7181   \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7182   \glossentrydesc{##1} & ##2\tabularnewline
7183   }%
```

Sub-entries on a separate row (no name, description in second column, page list in third column):

```
7184 \renewcommand{\subglossentry}[3]{%
7185   &
7186   \glssubentryitem{##2}%
7187   \glstarget{##2}{\strut}\glossentrydesc{##2} &
7188   ##3\tabularnewline
7189   }%
```

Blank row between groups:

```
7190 \renewcommand*{\glsgroupskip}{%
7191   \ifglsgnোগroupskip\else & &\tabularnewline\fi}%
7192 }
```

`longragged3colborder` The `longragged3colborder` style is like the `longragged3col` style but with a border:

```
7193 \newglossarystyle{longragged3colborder}{%
```

Base it on the `glostylelongragged3col` style:

```
7194 \setglossarystyle{longragged3col}%
```

Use a `longtable` with 3 columns with vertical lines around them:

```
7195 \renewenvironment{theglossary}%  
7196   {\begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|%  
7197     >{\raggedright}p{\glspagelistwidth}|}%  
7198   {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
7199 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%  
7200 }
```

`longragged3colheader` The `longragged3colheader` style is like `longragged3col` but with a header row:

```
7201 \newglossarystyle{longragged3colheader}{%
```

Base it on the `glostylelongragged3col` style:

```
7202 \setglossarystyle{longragged3col}%
```

Set the table's header:

```
7203 \renewcommand*{\glossaryheader}{%  
7204   \bfseries\entryname&\bfseries\descriptionname&  
7205   \bfseries\pagelistname\tabularnewline\endhead}%  
7206 }
```

`longragged3colheaderborder` The `longragged3colheaderborder` style is like the above but with a border

```
7207 \newglossarystyle{longragged3colheaderborder}{%
```

Base it on the `glostylelongragged3colborder` style:

```
7208 \setglossarystyle{longragged3colborder}%
```

Set the table's header and add horizontal line at table's foot:

```
7209 \renewcommand*{\glossaryheader}{%  
7210   \hline  
7211   \bfseries\entryname&\bfseries\descriptionname&  
7212   \bfseries\pagelistname\tabularnewline\hline\endhead  
7213   \hline\endfoot}%  
7214 }
```

`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.

```
7215 \newglossarystyle{altlongragged4col}{%
```

Use a `longtable` with 4 columns where the second and last columns may have multiple lines in each row:

```
7216 \renewenvironment{theglossary}%  
7217   {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%  
7218     >{\raggedright}p{\glspagelistwidth}}}%  
7219   {\end{longtable}}%
```

No table header:

```
7220 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7221 \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):

```
7222 \renewcommand{\glossentry}[2]{%
7223   \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7224   \glossentrydesc{##1} & \glossentrydesc{##1} &
7225   ##2\tabularnewline
7226 }%
```

Sub entries on a single row with no name (description in second column, symbol in third column, page list in last column):

```
7227 \renewcommand{\subglossentry}[3]{%
7228   &
7229   \glssubentryitem{##2}%
7230   \glstarget{##2}{\strut}\glossentrydesc{##2} &
7231   \glossentrysymbol{##2} & ##3\tabularnewline
7232 }%
```

Blank row between groups:

```
7233 \renewcommand*{\glsgroupskip}{%
7234   \ifglsgnোগroupskip\else & & \tabularnewline\fi}%
7235 }
```

`ongragged4colheader` The `altlongragged4colheader` style is like `altlongragged4col` but with a header row.

```
7236 \newglossarystyle{altlongragged4colheader}{%
```

Base it on the `glostylealtlongragged4col` style:

```
7237 \setglossarystyle{altlongragged4col}%
```

Use a `longtable` with 4 columns where the second and last columns may have multiple lines in each row:

```
7238 \renewenvironment{theglossary}%
7239   {\begin{longtable}{l>{\raggedright}p{\glsgdescwidth}l%
7240     >{\raggedright}p{\glspagelistwidth}}}%
7241   {\end{longtable}}%
```

Table has a header:

```
7242 \renewcommand*{\glossaryheader}{%
7243   \bfseries\entryname&\bfseries\descriptionname&
7244   \bfseries \symbolname&
7245   \bfseries\pagelistname\tabularnewline\endhead}%
7246 }
```

`ongragged4colborder` The `altlongragged4colborder` style is like `altlongragged4col` but with a border.

```
7247 \newglossarystyle{altlongragged4colborder}{%
```

Base it on the `glostylealtlongragged4col` style:

```
7248 \setglossarystyle{altlongragged4col}%
```

Use a `longtable` with 4 columns where the second and last columns may have multiple lines in each row:

```
7249 \renewenvironment{theglossary}%
7250 {\begin{longtable}{|l|>{\raggedright}p{\glstdescwidth}|l|}%
7251 >{\raggedright}p{\glspagelistwidth}|}%
7252 {\end{longtable}}%
```

Add horizontal lines to the head and foot of the table:

```
7253 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
7254 }
```

`ged4colheaderborder` The `altlongragged4colheaderborder` style is like the above but with a header as well as a border.

```
7255 \newglossarystyle{altlongragged4colheaderborder}{%
```

Base it on the `glostylealtlongragged4col` style:

```
7256 \setglossarystyle{altlongragged4col}%
```

Use a `longtable` with 4 columns where the second and last columns may have multiple lines in each row:

```
7257 \renewenvironment{theglossary}%
7258 {\begin{longtable}{|l|>{\raggedright}p{\glstdescwidth}|l|}%
7259 >{\raggedright}p{\glspagelistwidth}|}%
7260 {\end{longtable}}%
```

Add table header and horizontal line at the table's foot:

```
7261 \renewcommand*{\glossaryheader}{%
7262 \hline\bfseries\entryname&\bfseries\descriptionname&
7263 \bfseries \symbolname&
7264 \bfseries\pagelistname\tabularnewline\hline\endhead
7265 \hline\endfoot}%
7266 }
```

4.6 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.

```
7267 \ProvidesPackage{glossary-mcols}[2013/11/14 v4.0 (NLCT)]
```

Required packages:

```
7268 \RequirePackage{multicol}
7269 \RequirePackage{glossary-tree}
```

`\glsmcols` Define macro in which to store the number of columns. (Defaults to 2.)

```
7270 \newcommand*{\glsmcols}{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.)

```

7271 \newglossarystyle{mcolindex}{%
7272   \setglossarystyle{index}%
7273   \renewenvironment{theglossary}%
7274     {%
7275       \begin{multicols}{\glsmcols}
7276       \setlength{\parindent}{0pt}%
7277       \setlength{\parskip}{0pt plus 0.3pt}%
7278       \let\item\@idxitem}%
7279     {\end{multicols}}%
7280 }
```

mcolindexgroup As mcolindex but has headings:

```

7281 \newglossarystyle{mcolindexgroup}{%
7282   \setglossarystyle{mcolindex}%
7283   \renewcommand*{\glsgroupheading}[1]{%
7284     \item\textbf{\glsgroupheading{##1}}\indexspace}%
7285 }
```

mcolindexhypergroup The mcolindexhypergroup style is like the mcolindexgroup style but has hyper navigation.

```

7286 \newglossarystyle{mcolindexhypergroup}{%
  Base it on the glostylemcolindex style:
7287   \setglossarystyle{mcolindex}%
  Put navigation links to the groups at the start of the glossary:
7288   \renewcommand*{\glossaryheader}{%
7289     \item\textbf{\glsnavigation}\indexspace}%
  Add a heading for each group (with a target). The group's title is in bold followed
  by a vertical gap.
7290   \renewcommand*{\glsgroupheading}[1]{%
7291     \item\textbf{\glsnavigationhypertarget{##1}}{\glsgroupheading{##1}}}%
7292     \indexspace}%
7293 }
```

mcoltree Multi-column index style. Same as the tree, but puts the glossary in multiple columns.

```

7294 \newglossarystyle{mcoltree}{%
7295   \setglossarystyle{tree}%
7296   \renewenvironment{theglossary}%
7297     {%
7298       \begin{multicols}{\glsmcols}
7299       \setlength{\parindent}{0pt}%
7300       \setlength{\parskip}{0pt plus 0.3pt}%

```

```

7301 }%
7302 {\end{multicols}}}%
7303 }

```

mcoltreegroup Like the mcoltree style but the glossary groups have headings.

```

7304 \newglossarystyle{mcoltreegroup}{%
    Base it on the glostylemcoltree style:
7305 \setglossarystyle{mcoltree}%
    Each group has a heading (in bold) followed by a vertical gap):
7306 \renewcommand{\glsgroupheading}[1]{\par
7307 \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
7308 }

```

mcoltreehypergroup The mcoltreehypergroup style is like the treegroup style, but has a set of links to the groups at the start of the glossary.

```

7309 \newglossarystyle{mcoltreehypergroup}{%
    Base it on the glostylemcoltree style:
7310 \setglossarystyle{mcoltree}%
    Put navigation links to the groups at the start of the theglossary environment:
7311 \renewcommand*{\glossaryheader}{%
7312 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
    Each group has a heading (in bold with a target) followed by a vertical gap):
7313 \renewcommand*{\glsgroupheading}[1]{%
7314 \par\noindent
7315 \textbf{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}\par
7316 \indexspace}%
7317 }

```

mcoltreenoname Multi-column index style. Same as the treenoname, but puts the glossary in multiple columns.

```

7318 \newglossarystyle{mcoltreenoname}{%
7319 \setglossarystyle{treenoname}%
7320 \renewenvironment{theglossary}%
7321 {%
7322 \begin{multicols}{\glsmcols}
7323 \setlength{\parindent}{0pt}%
7324 \setlength{\parskip}{0pt plus 0.3pt}%
7325 }%
7326 {\end{multicols}}}%
7327 }

```

mcoltreenonamegroup Like the mcoltreenoname style but the glossary groups have headings.

```

7328 \newglossarystyle{mcoltreenonamegroup}{%
    Base it on the glostylemcoltreenoname style:
7329 \setglossarystyle{mcoltreenoname}%

```

Give each group a heading:

```
7330 \renewcommand{\glsgroupheading}[1]{\par
7331 \noindent\textbf{\glsgetgrouptitle{##1}}\par\indexspace}%
7332 }
```

`reenonamehypergroup` The `mcoltreenonamehypergroup` style is like the `mcoltreenonamegroup` style, but has a set of links to the groups at the start of the glossary.

```
7333 \newglossarystyle{mcoltreenonamehypergroup}{%
```

Base it on the `glostylemcoltreenoname` style:

```
7334 \setglossarystyle{mcoltreenoname}%
```

Put navigation links to the groups at the start of the `theglossary` environment:

```
7335 \renewcommand*{\glossaryheader}{%
7336 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
7337 \renewcommand*{\glsgroupheading}[1]{%
7338 \par\noindent
7339 \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
7340 \indexspace}%
7341 }
```

`mcolalttree` Multi-column index style. Same as the `alttree`, but puts the glossary in multiple columns.

```
7342 \newglossarystyle{mcolalttree}{%
7343 \setglossarystyle{alttree}%
7344 \renewenvironment{theglossary}%
7345 {%
7346 \begin{multicols}{\glsmcols}
7347 \def\@gls@prevlevel{-1}%
7348 \mbox{}\par
7349 }%
7350 {\par\end{multicols}}%
7351 }
```

`mcolalttreegroup` Like the `mcolalttree` style but the glossary groups have headings.

```
7352 \newglossarystyle{mcolalttreegroup}{%
```

Base it on the `glostylemcolalttree` style:

```
7353 \setglossarystyle{mcolalttree}%
```

Give each group a heading.

```
7354 \renewcommand{\glsgroupheading}[1]{\par
7355 \def\@gls@prevlevel{-1}%
7356 \hangindent0pt\relax
7357 \parindent0pt\relax
7358 \textbf{\glsgetgrouptitle{##1}}\par\indexspace}%
7359 }
```

`colalttreehypergroup` The `colalttreehypergroup` style is like the `colalttreegroup` style, but has a set of links to the groups at the start of the glossary.

```
7360 \newglossarystyle{colalttreehypergroup}{%
```

Base it on the `glostymcolalttree` style:

```
7361 \setglossarystyle{colalttree}{%
```

Put the navigation links in the header

```
7362 \renewcommand*{\glossaryheader}{%
```

```
7363 \par
```

```
7364 \def\@gls@prevlevel{-1}%
```

```
7365 \hangindent0pt\relax
```

```
7366 \parindent0pt\relax
```

```
7367 \textbf{\glsnavigation}\par\indexspace}%
```

Put a hypertarget at the start of each group

```
7368 \renewcommand*{\glsgroupheading}[1]{%
```

```
7369 \par
```

```
7370 \def\@gls@prevlevel{-1}%
```

```
7371 \hangindent0pt\relax
```

```
7372 \parindent0pt\relax
```

```
7373 \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
```

```
7374 \indexspace}}
```

4.7 Glossary Styles using supertabular environment (glossary-super package)

The glossary styles defined in the package use the supertabular environment.

```
7375 \ProvidesPackage{glossary-super}[2013/11/14 v4.0 (NLCT)]
```

Requires the package:

```
7376 \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.

```
7377 \@ifundefined{glsdescwidth}{%
```

```
7378 \newlength\glsdescwidth
```

```
7379 \setlength{\glsdescwidth}{0.6\hsize}
```

```
7380 }{}}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined if has been loaded.

```
7381 \@ifundefined{glspagelistwidth}{%
```

```
7382 \newlength\glspagelistwidth
```

```
7383 \setlength{\glspagelistwidth}{0.1\hsize}
```

```
7384 }{}}
```

`super` The `super` glossary style uses the supertabular environment (it uses lengths defined in the package.)

```
7385 \newglossarystyle{super}{%
```

Put the glossary in a supertabular environment with two columns and no head or tail:

```
7386 \renewenvironment{theglossary}%
7387   {\tablehead{}\tabletail{}}%
7388   \begin{supertabular}{lp{\glsdescwidth}}%
7389   {\end{supertabular}}%
```

Do nothing at the start of the table:

```
7390 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7391 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries put in a row (name in first column, description and page list in second column):

```
7392 \renewcommand{\glossentry}[2]{%
7393   \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7394   \glossentrydesc{##1}\glspostdescription\space ##2\tabularnewline
7395   }%
```

Sub entries put in a row (no name, description and page list in second column):

```
7396 \renewcommand{\subglossentry}[3]{%
7397   &
7398   \glssubentryitem{##2}%
7399   \glstarget{##2}{\strut}\glosentrydesc{##2}\glspostdescription\space
7400   ##3\tabularnewline
7401   }%
```

Blank row between groups:

```
7402 \renewcommand*{\glsgroupskip}{}%
7403   \ifglsnogroupskip\else & \tabularnewline\fi}%
7404 }
```

superborder The superborder style is like the above, but with horizontal and vertical lines:

```
7405 \newglossarystyle{superborder}{}%
```

Base it on the glostylesuper style:

```
7406 \setglossarystyle{super}{}%
```

Put the glossary in a supertabular environment with two columns and a horizontal line in the head and tail:

```
7407 \renewenvironment{theglossary}%
7408   {\tablehead{\hline}\tabletail{\hline}%
7409   \begin{supertabular}{|lp{\glsdescwidth}|}%
7410   {\end{supertabular}}%
7411 }
```

superheader The superheader style is like the super style, but with a header:

```
7412 \newglossarystyle{superheader}{}%
```

Base it on the glostylesuper style:

```
7413 \setglossarystyle{super}{}%
```

Put the glossary in a supertabular environment with two columns, a header and no tail:

```

7414 \renewenvironment{theglossary}%
7415   {\tablehead{\bfseries \entryname &
7416     \bfseries \descriptionname\tabularnewline}%
7417     \tabletail{}}%
7418   \begin{supertabular}{lp{\glsdescwidth}}}%
7419   {\end{supertabular}}}%
7420 }
```

superheaderborder The superheaderborder style is like the super style but with a header and border:

```
7421 \newglossarystyle{superheaderborder}{%
```

Base it on the glostylesuper style:

```
7422   \setglossarystyle{super}%
```

Put the glossary in a supertabular environment with two columns, a header and horizontal lines above and below the table:

```

7423   \renewenvironment{theglossary}%
7424     {\tablehead{\hline\bfseries \entryname &
7425       \bfseries \descriptionname\tabularnewline\hline}%
7426       \tabletail{\hline}
7427       \begin{supertabular}{|lp{\glsdescwidth}|}%
7428       {\end{supertabular}}}%
7429 }
```

super3col The super3col style is like the super style, but with 3 columns:

```
7430 \newglossarystyle{super3col}{%
```

Put the glossary in a supertabular environment with three columns and no head or tail:

```

7431   \renewenvironment{theglossary}%
7432     {\tablehead{}\tabletail{}}%
7433     \begin{supertabular}{lp{\glsdescwidth}p{\glspagelistwidth}}}%
7434     {\end{supertabular}}}%

```

Do nothing at the start of the table:

```
7435   \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7436   \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```

7437   \renewcommand{\glossentry}[2]{%
7438     \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7439     \glossentrydesc{##1} & ##2\tabularnewline
7440   }%
```

Sub entries on a row (no name, description in second column, page list in last column):

```

7441 \renewcommand{\subglossentry}[3]{%
7442     &
7443     \glssubentryitem{##2}%
7444     \glstarget{##2}{\strut}\glossentrydesc{##2} &
7445     ##3\tabularnewline
7446 }%
```

Blank row between groups:

```

7447 \renewcommand*{\glsgroupskip}{%
7448     \ifglsgnogroupskip\else & &\tabularnewline\fi}%
7449 }
```

super3colborder The super3colborder style is like the super3col style, but with a border:

```

7450 \newglossarystyle{super3colborder}{%
    Base it on the glostylesuper3col style:
7451     \setglossarystyle{super3col}%
    Put the glossary in a supertabular environment with three columns and a horizontal line in the head and tail:
7452     \renewenvironment{theglossary}%
7453         {\tablehead{\hline}\tabletail{\hline}%
7454         \begin{supertabular}{|l|p{\glsgdescwidth}|p{\glspagelistwidth}|}%
7455         {\end{supertabular}}%
7456 }
```

super3colheader The super3colheader style is like the super3col style but with a header row:

```

7457 \newglossarystyle{super3colheader}{%
    Base it on the glostylesuper3col style:
7458     \setglossarystyle{super3col}%
    Put the glossary in a supertabular environment with three columns, a header and no tail:
7459     \renewenvironment{theglossary}%
7460         {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
7461             \bfseries\pagelistname\tabularnewline}\tabletail{}}%
7462         \begin{supertabular}{lp{\glsgdescwidth}p{\glspagelistwidth}}%
7463         {\end{supertabular}}%
7464 }
```

super3colheaderborder The super3colheaderborder style is like the super3col style but with a header and border:

```

7465 \newglossarystyle{super3colheaderborder}{%
    Base it on the glostylesuper3colborder style:
7466     \setglossarystyle{super3colborder}%

```

Put the glossary in a supertabular environment with three columns, a header with horizontal lines and a horizontal line in the tail:

```

7467 \renewenvironment{theglossary}%
7468   {\tablehead{\hline
7469     \bfseries\entryname&\bfseries\descriptionname&
7470     \bfseries\pagelistname\tabularnewline\hline}%
7471   \tabletail{\hline}%
7472   \begin{supertabular}{|l|p{\glstdescwidth}|p{\glspagelistwidth}|}%
7473   {\end{supertabular}}%
7474 }

```

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.

```

7475 \newglossarystyle{super4col}{%

```

Put the glossary in a supertabular environment with four columns and no head or tail:

```

7476 \renewenvironment{theglossary}%
7477   {\tablehead{}\tabletail{}}%
7478   \begin{supertabular}{|l|l|l|l|}%
7479   \end{supertabular}}%

```

Do nothing at the start of the table:

```

7480 \renewcommand*{\glossaryheader}{}%

```

No group headings:

```

7481 \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:

```

7482 \renewcommand{\glossentry}[2]{%
7483   \glssentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7484   \glossentrydesc{##1} &
7485   \glossentrysymbol{##1} & ##3\tabularnewline
7486 }%

```

Sub entries on a row with no name, the description in the second column, symbol in third column and page list in last column:

```

7487 \renewcommand{\subglossentry}[3]{%
7488   &
7489   \glssubentryitem{##2}%
7490   \glstarget{##2}{\strut}\glossentrydesc{##2} &
7491   \glossentrysymbol{##2} & ##3\tabularnewline
7492 }%

```

Blank row between groups:

```

7493 \renewcommand*{\glsgroupskip}{%
7494   \ifglsgroupskip\else & & \tabularnewline\fi}%
7495 }

```

super4colheader The super4colheader style is like the super4col but with a header row.

```
7496 \newglossarystyle{super4colheader}{%
    Base it on the glostylesuper4col style:
7497 \setglossarystyle{super4col}%
    Put the glossary in a supertabular environment with four columns, a header and
    no tail:
7498 \renewenvironment{theglossary}%
7499     {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
7500                \bfseries\symbolname &
7501                \bfseries\pagelistname\tabularnewline}%
7502     \tabletail{}}%
7503     \begin{supertabular}{|l|l|l|l|}%
7504     {\end{supertabular}}%
7505 }
```

super4colborder The super4colborder style is like the super4col but with a border.

```
7506 \newglossarystyle{super4colborder}{%
    Base it on the glostylesuper4col style:
7507 \setglossarystyle{super4col}%
    Put the glossary in a supertabular environment with four columns and a hori-
    zontal line in the head and tail:
7508 \renewenvironment{theglossary}%
7509     {\tablehead{\hline}\tabletail{\hline}%
7510     \begin{supertabular}{|l|l|l|l|}%
7511     {\end{supertabular}}%
7512 }
```

super4colheaderborder The super4colheaderborder style is like the super4col but with a header and border.

```
7513 \newglossarystyle{super4colheaderborder}{%
    Base it on the glostylesuper4col style:
7514 \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:
7515 \renewenvironment{theglossary}%
7516     {\tablehead{\hline\bfseries\entryname&\bfseries\descriptionname&
7517                \bfseries\symbolname &
7518                \bfseries\pagelistname\tabularnewline\hline}%
7519     \tabletail{\hline}%
7520     \begin{supertabular}{|l|l|l|l|}%
7521     {\end{supertabular}}%
7522 }
```

`altsuper4col` The `altsuper4col` glossary style is like `super4col` but has provision for multiline descriptions.

```
7523 \newglossarystyle{altsuper4col}{%
    Base it on the glostylesuper4col style:
7524   \setglossarystyle{super4col}%

    Put the glossary in a supertabular environment with four columns and no head
    or tail:
7525   \renewenvironment{theglossary}%
7526     {\tablehead{}\tabletail{}}%
7527     \begin{supertabular}{lp{\glsgdescwidth}lp{\glspagelistwidth}}}%
7528     {\end{supertabular}}}%
7529 }
```

`altsuper4colheader` The `altsuper4colheader` style is like the `altsuper4col` but with a header row.

```
7530 \newglossarystyle{altsuper4colheader}{%
    Base it on the glostylesuper4colheader style:
7531   \setglossarystyle{super4colheader}%

    Put the glossary in a supertabular environment with four columns, a header and
    no tail:
7532   \renewenvironment{theglossary}%
7533     {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
7534               \bfseries\symbolname &
7535               \bfseries\pagelistname\tabularnewline}\tabletail{}}%
7536     \begin{supertabular}{lp{\glsgdescwidth}lp{\glspagelistwidth}}}%
7537     {\end{supertabular}}}%
7538 }
```

`altsuper4colborder` The `altsuper4colborder` style is like the `altsuper4col` but with a border.

```
7539 \newglossarystyle{altsuper4colborder}{%
    Base it on the glostylesuper4colborder style:
7540   \setglossarystyle{super4colborder}%

    Put the glossary in a supertabular environment with four columns and a hori-
    zontal line in the head and tail:
7541   \renewenvironment{theglossary}%
7542     {\tablehead{\hline}\tabletail{\hline}%
7543     \begin{supertabular}%
7544       {\lllp{\glsgdescwidth}lllp{\glspagelistwidth}}}%
7545     {\end{supertabular}}}%
7546 }
```

`per4colheaderborder` The `altsuper4colheaderborder` style is like the `altsuper4col` but with a header and border.

```
7547 \newglossarystyle{altsuper4colheaderborder}{%
```

Base it on the `glostylessuper4colheaderborder` style:

```
7548 \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:

```
7549 \renewenvironment{theglossary}%
7550   {\tablehead{\hline
7551     \bfseries\entryname &
7552     \bfseries\descriptionname &
7553     \bfseries\symbolname &
7554     \bfseries\pagelistname\tabularnewline\hline}%
7555   \tabletail{\hline}%
7556   \begin{supertabular}%
7557     {\lllp{\glstdescwidth}\lllp{\glspagelistwidth}}}%
7558   {\end{supertabular}}%
7559 }
```

4.8 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.

```
7560 \ProvidesPackage{glossary-superragged}[2013/11/14 v4.0 (NLCT)]
```

Requires the package:

```
7561 \RequirePackage{array}
```

Requires the package:

```
7562 \RequirePackage{supertabular}
```

`\glstdescwidth` This is a length that governs the width of the description column. This may already have been defined.

```
7563 \@ifundefined{glstdescwidth}{%
7564   \newlength\glstdescwidth
7565   \setlength{\glstdescwidth}{0.6\hsize}
7566 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

```
7567 \@ifundefined{glspagelistwidth}{%
7568   \newlength\glspagelistwidth
7569   \setlength{\glspagelistwidth}{0.1\hsize}
7570 }{}
```

`superragged` The `superragged` glossary style uses the `supertabular` environment.

```
7571 \newglossarystyle{superragged}{%
```

Put the glossary in a supertabular environment with two columns and no head or tail:

```
7572 \renewenvironment{theglossary}%
7573   {\tablehead{}\tabletail{}}%
7574   \begin{supertabular}{|l>{\raggedright}p{\glsgdescwidth}}}%
7575   {\end{supertabular}}%
```

Do nothing at the start of the table:

```
7576 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7577 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries put in a row (name in first column, description and page list in second column):

```
7578 \renewcommand{\glossentry}[2]{%
7579   \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7580   \glossentrydesc{##1}\glspostdescription\space ##2%
7581   \tabularnewline
7582 }%
```

Sub entries put in a row (no name, description and page list in second column):

```
7583 \renewcommand{\subglossentry}[3]{%
7584   &
7585   \glssubentryitem{##2}%
7586   \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space
7587   ##3%
7588   \tabularnewline
7589 }%
```

Blank row between groups:

```
7590 \renewcommand*{\glsgroupskip}{\ifglsgnigroupskip\else & \tabularnewline\fi}%
7591 }
```

superraggedborder The superraggedborder style is like the above, but with horizontal and vertical lines:

```
7592 \newglossarystyle{superraggedborder}{%
```

Base it on the glostylessuperragged style:

```
7593 \setglossarystyle{superragged}%
```

Put the glossary in a supertabular environment with two columns and a horizontal line in the head and tail:

```
7594 \renewenvironment{theglossary}%
7595   {\tablehead{\hline}\tabletail{\hline}}%
7596   \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}}|}%
7597   {\end{supertabular}}%
7598 }
```

superraggedheader The superraggedheader style is like the super style, but with a header:

```
7599 \newglossarystyle{superraggedheader}{%
```

Base it on the `glostylesuperragged` style:

```
7600 \setglossarystyle{superragged}%
```

Put the glossary in a `supertabular` environment with two columns, a header and no tail:

```
7601 \renewenvironment{theglossary}%  
7602 {\tablehead{\bfseries \entryname & \bfseries \descriptionname  
7603 \tabularnewline}%  
7604 \tabletail{}}%  
7605 \begin{supertabular}{1>\raggedright}p{\glsgdescwidth}}}%  
7606 {\end{supertabular}}%  
7607 }
```

`rraggedheaderborder` The `superraggedheaderborder` style is like the `superragged` style but with a header and border:

```
7608 \newglossarystyle{superraggedheaderborder}{%
```

Base it on the `glostylesuper` style:

```
7609 \setglossarystyle{superragged}%
```

Put the glossary in a `supertabular` environment with two columns, a header and horizontal lines above and below the table:

```
7610 \renewenvironment{theglossary}%  
7611 {\tablehead{\hline\bfseries \entryname &  
7612 \bfseries \descriptionname\tabularnewline\hline}%  
7613 \tabletail{\hline}  
7614 \begin{supertabular}{|1|>\raggedright}p{\glsgdescwidth}|}}}%  
7615 {\end{supertabular}}%  
7616 }
```

`superragged3col` The `superragged3col` style is like the `superragged` style, but with 3 columns:

```
7617 \newglossarystyle{superragged3col}{%
```

Put the glossary in a `supertabular` environment with three columns and no head or tail:

```
7618 \renewenvironment{theglossary}%  
7619 {\tablehead{} \tabletail{}}%  
7620 \begin{supertabular}{1>\raggedright}p{\glsgdescwidth}%  
7621 >\raggedright}p{\glspagelistwidth}}}%  
7622 {\end{supertabular}}%
```

Do nothing at the start of the table:

```
7623 \renewcommand*\glossaryheader{}%
```

No group headings:

```
7624 \renewcommand*\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
7625 \renewcommand{\glossentry}[2]{}%  
7626 \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
```

```

7627 \glossentrydesc{##1} &
7628 ##2\tabularnewline
7629 }%

```

Sub entries on a row (no name, description in second column, page list in last column):

```

7630 \renewcommand{\subglossentry}[3]{%
7631 &
7632 \glssubentryitem{##2}%
7633 \glstarget{##2}{\strut}\glossentrydesc{##2} &
7634 ##3\tabularnewline
7635 }%

```

Blank row between groups:

```

7636 \renewcommand*{\glsgroupskip}{\ifglsgnogroupskip\else & &\tabularnewline\fi}%
7637 }

```

superragged3colborder The `superragged3colborder` style is like the `superragged3col` style, but with a border:

```

7638 \newglossarystyle{superragged3colborder}{%

```

Base it on the `glostylesuperragged3col` style:

```

7639 \setglossarystyle{superragged3col}%

```

Put the glossary in a `supertabular` environment with three columns and a horizontal line in the head and tail:

```

7640 \renewenvironment{theglossary}%
7641 {\tablehead{\hline}\tabletail{\hline}%
7642 \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}|%
7643 >{\raggedright}p{\glspagelistwidth}|}%
7644 {\end{supertabular}}%
7645 }

```

superragged3colheader The `superragged3colheader` style is like the `superragged3col` style but with a header row:

```

7646 \newglossarystyle{superragged3colheader}{%

```

Base it on the `glostylesuperragged3col` style:

```

7647 \setglossarystyle{superragged3col}%

```

Put the glossary in a `supertabular` environment with three columns, a header and no tail:

```

7648 \renewenvironment{theglossary}%
7649 {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
7650 \bfseries\pagelistname\tabularnewline}\tabletail{}}%
7651 \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}%
7652 >{\raggedright}p{\glspagelistwidth}}}%
7653 {\end{supertabular}}%
7654 }

```

ght3colheaderborder The superragged3colheaderborder style is like the superragged3col style but with a header and border:

```
7655 \newglossarystyle{superragged3colheaderborder}{%
```

Base it on the glostylesuperragged3colborder style:

```
7656 \setglossarystyle{superragged3colborder}{%
```

Put the glossary in a supertabular environment with three columns, a header with horizontal lines and a horizontal line in the tail:

```
7657 \renewenvironment{theglossary}{%
7658   {\tablehead{\hline
7659     \bfseries\entryname&\bfseries\descriptionname&
7660     \bfseries\pagelistname\tabularnewline\hline}%
7661   \tabletail{\hline}%
7662   \begin{supertabular}{|l|>{\raggedright}p{\glsdescwidth}|%
7663     >{\raggedright}p{\glspagelistwidth}|}%
7664   {\end{supertabular}}}%
7665 }
```

altsuperragged4col The altsuperragged4col glossary style is like altsuper4col style in the package but uses ragged right formatting in the description and page list columns.

```
7666 \newglossarystyle{altsuperragged4col}{%
```

Put the glossary in a supertabular environment with four columns and no head or tail:

```
7667 \renewenvironment{theglossary}{%
7668   {\tablehead{}\tabletail}%
7669   \begin{supertabular}{|l>{\raggedright}p{\glsdescwidth}l%
7670     >{\raggedright}p{\glspagelistwidth}}}%
7671   {\end{supertabular}}%
```

Do nothing at the start of the table:

```
7672 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7673 \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:

```
7674 \renewcommand{\glossentry}[2]{%
7675   \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7676   \glossentrydesc{##1} &
7677   \glossentrysymbol{##1} & ##2\tabularnewline
7678 }%
```

Sub entries on a row with no name, the description in the second column, symbol in third column and page list in last column:

```
7679 \renewcommand{\subglossentry}[3]{%
7680   &
7681   \glssubentryitem{##2}%
7682   \glstarget{##2}{\strut}\glossentrydesc{##2} &
```

```

7683     \glossentrysymbol{##2} & ##3\tabularnewline
7684 }%

```

Blank row between groups:

```

7685 \renewcommand*{\glsgroupskip}{\ifglsgnোগroupskip\else & & \tabularnewline\fi}%
7686 }

```

altperragged4colheader The altperragged4colheader style is like the altperragged4col style but with a header row.

```

7687 \newglossarystyle{altperragged4colheader}{%

```

Base it on the glostylealtperragged4col style:

```

7688 \setglossarystyle{altperragged4col}%

```

Put the glossary in a supertabular environment with four columns, a header and no tail:

```

7689 \renewenvironment{theglossary}%
7690 { \tablehead{\bfseries\entryname&\bfseries\descriptionname&
7691 \bfseries\symbolname &
7692 \bfseries\pagelistname\tabularnewline}\tabletail{}}%
7693 \begin{supertabular}{1>{\raggedright}p{\glsgdescwidth}1%
7694 >{\raggedright}p{\glspagelistwidth}}}%
7695 {\end{supertabular}}%
7696 }

```

altperragged4colborder The altperragged4colborder style is like the altperragged4col style but with a border.

```

7697 \newglossarystyle{altperragged4colborder}{%

```

Base it on the glostylealtperragged4col style:

```

7698 \setglossarystyle{altperragged4col}%

```

Put the glossary in a supertabular environment with four columns and a horizontal line in the head and tail:

```

7699 \renewenvironment{theglossary}%
7700 { \tablehead{\hline}\tabletail{\hline}%
7701 \begin{supertabular}%
7702 { |1|>{\raggedright}p{\glsgdescwidth}|1|%
7703 >{\raggedright}p{\glspagelistwidth}|}%
7704 {\end{supertabular}}%
7705 }

```

altperragged4colheaderborder The altperragged4colheaderborder style is like the altperragged4col style but with a header and border.

```

7706 \newglossarystyle{altperragged4colheaderborder}{%

```

Base it on the glostylealtperragged4col style:

```

7707 \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:

```

7708 \renewenvironment{theglossary}%
7709   {\tablehead{\hline
7710     \bfseries\entryname &
7711     \bfseries\descriptionname &
7712     \bfseries\symbolname &
7713     \bfseries\pagelistname\tabularnewline\hline}%
7714   \tabletail{\hline}%
7715   \begin{supertabular}%
7716     {|||>{\raggedright}p{\glsgdescwidth}|l|}%
7717     >{\raggedright}p{\glspagelistwidth}|}}%
7718   {\end{supertabular}}%
7719 }

```

4.9 Tree Styles (glossary-tree.sty)

The style file defines glossary styles that have a tree-like structure. These are designed for hierarchical glossaries.

```

7720 \ProvidesPackage{glossary-tree}[2013/11/14 v4.0 (NLCT)]

```

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.

```

7721 \newglossarystyle{index}{%

```

Set the paragraph indentation and skip and define `\item` to be the same as that used by the index:

```

7722 \renewenvironment{theglossary}%
7723   {\setlength{\parindent}{0pt}%
7724   \setlength{\parskip}{0pt plus 0.3pt}%
7725   \let\item\@idxitem}%
7726   {\par}%

```

Do nothing at the start of the environment:

```

7727 \renewcommand*{\glossaryheader}{}%

```

No group headers:

```

7728 \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.

```

7729 \renewcommand*{\glossentry}[2]{%
7730   \item\glsglentryitem{##1}\textbf{\glstarget{##1}{\glossentryname{##1}}}%
7731   \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
7732   \space \glossentrydesc{##1}\glspostdescription\space ##2%
7733 }%

```

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.

```

7734 \renewcommand{\subglossentry}[3]{%
7735   \ifcase##1\relax
7736     % level 0
7737     \item
7738   \or
7739     % level 1
7740     \subitem
7741     \glssubentryitem{##2}%
7742   \else
7743     % all other levels
7744     \subsubitem
7745   \fi
7746   \textbf{\glstarget{##2}{\glossentryname{##2}}}%
7747   \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
7748   \space\glossentrydesc{##2}\glspostdescription\space ##3%
7749 }%
```

Vertical gap between groups is the same as that used by indices:

```

7750 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}}
```

indexgroup The `indexgroup` style is like the `index` style but has headings.

```

7751 \newglossarystyle{indexgroup}{%
```

Base it on the `glostyleindex` style:

```

7752 \setglossarystyle{index}%
```

Add a heading for each group. This puts the group's title in bold followed by a vertical gap.

```

7753 \renewcommand*{\glsgroupheading}[1]{%
7754   \item\textbf{\glsgrouptitle{##1}}\indexspace}%
7755 }
```

indexhypergroup The `indexhypergroup` style is like the `indexgroup` style but has hyper navigation.

```

7756 \newglossarystyle{indexhypergroup}{%
```

Base it on the `glostyleindex` style:

```

7757 \setglossarystyle{index}%
```

Put navigation links to the groups at the start of the glossary:

```

7758 \renewcommand*{\glossaryheader}{%
7759   \item\textbf{\glshnavigation}\indexspace}%
```

Add a heading for each group (with a target). The group's title is in bold followed by a vertical gap.

```

7760 \renewcommand*{\glsgroupheading}[1]{%
7761   \item\textbf{\glshnavhypertarget{##1}{\glsgrouptitle{##1}}}%
7762 }
```

```

7762 \indexspace}%
7763 }

```

tree The tree glossary style is similar in style to the index style, but can have arbitrary levels.

```

7764 \newglossarystyle{tree}{%

```

Set the paragraph indentation and skip:

```

7765 \renewenvironment{theglossary}%
7766 {\setlength{\parindent}{0pt}%
7767 \setlength{\parskip}{0pt plus 0.3pt}}%
7768 {}%

```

Do nothing at the start of the theglossary environment:

```

7769 \renewcommand*{\glossaryheader}{}%

```

No group headings:

```

7770 \renewcommand*{\glsgroupheading}[1]{}%

```

Main (level 0) entries: name in bold, followed by symbol in brackets (if it exists), the description and the page list:

```

7771 \renewcommand{\glossentry}[2]{%
7772 \hangindent0pt\relax
7773 \parindent0pt\relax
7774 \glstryitem{##1}\textbf{\glstarget{##1}{\glossentryname{##1}}}%
7775 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
7776 \space\glossentrydesc{##1}\glspostdescription\space##2\par
7777 }%

```

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.

```

7778 \renewcommand{\subglossentry}[3]{%
7779 \hangindent##1\glstreeindent\relax
7780 \parindent##1\glstreeindent\relax
7781 \ifnum##1=1\relax
7782 \glssubentryitem{##2}%
7783 \fi
7784 \textbf{\glstarget{##2}{\glossentryname{##2}}}%
7785 \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
7786 \space\glossentrydesc{##2}\glspostdescription\space ##3\par
7787 }%

```

Vertical gap between groups is the same as that used by indices:

```

7788 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}}

```

treegroup Like the tree style but the glossary groups have headings.

```

7789 \newglossarystyle{treegroup}{%

```

Base it on the glostyletree style:

```

7790 \setglossarystyle{tree}%

```

Each group has a heading (in bold) followed by a vertical gap):

```
7791 \renewcommand{\glsgroupheading}[1]{\par
7792 \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
7793 }
```

treehypergroup The treehypergroup style is like the treegroup style, but has a set of links to the groups at the start of the glossary.

```
7794 \newglossarystyle{treehypergroup}{%
```

Base it on the glostyletree style:

```
7795 \setglossarystyle{tree}%
```

Put navigation links to the groups at the start of the theglossary environment:

```
7796 \renewcommand*\glossaryheader}{%
7797 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
7798 \renewcommand*\glsgroupheading}[1]{%
7799 \par\noindent
7800 \textbf{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}\par
7801 \indexspace}%
7802 }
```

\glstreeindent Length governing left indent for each level of the tree style.

```
7803 \newlength\glstreeindent
7804 \setlength{\glstreeindent}{10pt}
```

treenoname The treenoname glossary style is like the tree style, but doesn't print the name or symbol for sub-levels.

```
7805 \newglossarystyle{treenoname}{%
```

Set the paragraph indentation and skip:

```
7806 \renewenvironment{theglossary}%
7807 {\setlength{\parindent}{0pt}%
7808 \setlength{\parskip}{0pt plus 0.3pt}}%
7809 {}%
```

No header:

```
7810 \renewcommand*\glossaryheader}{}%
```

No group headings:

```
7811 \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.

```
7812 \renewcommand{\glossentry}[2]{%
7813 \hangindent0pt\relax
7814 \parindent0pt\relax
7815 \glstentryitem{##1}\textbf{\glstarget{##1}{\glossentryname{##1}}}%
7816 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
7817 \space\glossentrydesc{##1}\glspostdescription\space##2\par
7818 }%
```

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.

```
7819 \renewcommand{\subglossentry}[3]{%
7820   \hangindent##1\glstreeindent\relax
7821   \parindent##1\glstreeindent\relax
7822   \ifnum##1=1\relax
7823     \glssubentryitem{##2}%
7824   \fi
7825   \glstarget{##2}{\strut}%
7826   \glossentrydesc{##2}\glspostdescription\space##3\par
7827 }
```

Vertical gap between groups is the same as that used by indices:

```
7828 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
7829 }
```

`treenonamegroup` Like the `treenoname` style but the glossary groups have headings.

```
7830 \newglossarystyle{treenonamegroup}{%
```

Base it on the `glostyletreenoname` style:

```
7831 \setglossarystyle{treenoname}%
```

Give each group a heading:

```
7832 \renewcommand{\glsgroupheading}[1]{\par
7833   \noindent\textbf{\glsggetgrouptitle{##1}}\par\indexspace}%
7834 }
```

`treenonamehypergroup` The `treenonamehypergroup` style is like the `treenonamegroup` style, but has a set of links to the groups at the start of the glossary.

```
7835 \newglossarystyle{treenonamehypergroup}{%
```

Base it on the `glostyletreenoname` style:

```
7836 \setglossarystyle{treenoname}%
```

Put navigation links to the groups at the start of the `theglossary` environment:

```
7837 \renewcommand*{\glossaryheader}{%
7838   \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
7839 \renewcommand*{\glsgroupheading}[1]{%
7840   \par\noindent
7841   \textbf{\glsnavhypertarget{##1}{\glsggetgrouptitle{##1}}}\par
7842   \indexspace}%
7843 }
```

`\glsssetwidest` `\glsssetwidest[$\langle level \rangle$]{ $\langle text \rangle$ }` sets the widest text for the given level. It is used by the `alttree` glossary styles to determine the indentation of each level.

```
7844 \newcommand*{\glsssetwidest}[2][0]{%
7845   \expandafter\def\csname @glswidestname\romannumeral#1\endcsname{%
7846     #2}%
7847 }
```

```

\@glswidestname Initialise \@glswidestname.
7848 \newcommand*{\@glswidestname}{%

almtree The almtree glossary style is similar in style to the tree style, but the inden-
        tation is obtained from the width of \@glswidestname which is set using
        \glsetwidest.
7849 \newglossarystyle{almtree}{%
        Redefine theglossary environment.
7850 \renewenvironment{theglossary}%
7851     {\def\@gls@prevlevel{-1}%
7852      \mbox{}\par}%
7853     {\par}%

        Set the header and group headers to nothing.
7854 \renewcommand*{\glossaryheader}{}%
7855 \renewcommand*{\glsgroupheading}[1]{}%

        Redefine the way that the level 0 entries are displayed.
7856 \renewcommand{\glossentry}[2]{%

        If the level hasn't changed, keep the same settings, otherwise change \glstreeindent
        accordingly.
7857 \ifnum\@gls@prevlevel=0\relax
7858 \else

        Find out how big the indentation should be by measuring the widest entry.
7859 \settowidth{\glstreeindent}{\textbf{\@glswidestname\space}}%

        Set the hangindent and paragraph indent.
7860 \hangindent\glstreeindent
7861 \parindent\glstreeindent
7862 \fi

        Put the name to the left of the paragraph block.
7863 \makebox[0pt][r]{\makebox[\glstreeindent][l]{%
7864 \glsentryitem{##1}\textbf{\glstarget{##1}{\glossentryname{##1}}}}}%

        If the symbol is missing, ignore it, otherwise put it in brackets.
7865 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%

        Do the description followed by the description terminator and location list.
7866 \glossentrydesc{##1}\glspostdescription \space ##2\par

        Set the previous level to 0.
7867 \def\@gls@prevlevel{0}%
7868 }%

        Redefine the way sub-entries are displayed.
7869 \renewcommand{\subglossentry}[3]{%

```

Increment and display the sub-entry counter if this is a level 1 entry and the sub-entry counter is in use.

```
7870    \ifnum##1=1\relax
7871      \glssubentryitem{##2}%
7872    \fi
```

If the level hasn't changed, keep the same settings, otherwise adjust `\glstreeindent` accordingly.

```
7873    \ifnum\@gls@prevlevel=##1\relax
7874    \else
```

Compute the widest entry for this level, or for level 0 if not defined for this level.

Store in `\gls@tmplen`

```
7875      \@ifundefined{@glswidestname\romannumeral##1}{%
7876        \settowidth{\gls@tmplen}{\textbf{@glswidestname\space}}{%
7877        \settowidth{\gls@tmplen}{\textbf{%
7878          \csname @glswidestname\romannumeral##1\endcsname\space}}}%

```

Determine if going up or down a level

```
7879      \ifnum\@gls@prevlevel<##1\relax
```

Depth has increased, so add the width of the widest entry to `\glstreeindent`.

```
7880        \setlength\glstreeindent\gls@tmplen
7881        \addtolength\glstreeindent\parindent
7882        \parindent\glstreeindent
7883      \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`.

```
7884        \@ifundefined{@glswidestname\romannumeral\@gls@prevlevel}{%
7885          \settowidth{\glstreeindent}{\textbf{%
7886            @glswidestname\space}}{%
7887          \settowidth{\glstreeindent}{\textbf{%
7888            \csname @glswidestname\romannumeral\@gls@prevlevel
7889              \endcsname\space}}}%

```

Subtract this length from the previous level's paragraph indent and set to `\glstreeindent`.

```
7890        \addtolength\parindent{-\glstreeindent}%
7891        \setlength\glstreeindent\parindent
7892      \fi
7893    \fi
```

Set the hanging indentation.

```
7894    \hangindent\glstreeindent
```

Put the name to the left of the paragraph block

```
7895    \makebox[0pt][r]{\makebox[\gls@tmplen][l]{%
7896      \textbf{\glstarget{##2}{\glossentryname{##2}}}}}%

```

If the symbol is missing, ignore it, otherwise put it in brackets.

```
7897 \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}}%
```

Do the description followed by the description terminator and location list.

```
7898 \glossentrydesc{##2}\glspostdescription\space ##3\par
```

Set the previous level macro to the current level.

```
7899 \def\@gls@prevlevel{##1}%  
7900 }%
```

Vertical gap between groups is the same as that used by indices:

```
7901 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%  
7902 }
```

almtreegroup Like the almtree style but the glossary groups have headings.

```
7903 \newglossarystyle{almtreegroup}{%
```

Base it on the glostylealmtree style:

```
7904 \setglossarystyle{almtree}%
```

Give each group a heading.

```
7905 \renewcommand{\glsgroupheading}[1]{\par  
7906 \def\@gls@prevlevel{-1}%  
7907 \hangindent0pt\relax  
7908 \parindent0pt\relax  
7909 \textbf{\glsgetgrouptitle{##1}}\par\indexspace}%  
7910 }
```

almtreehypergroup The almtreehypergroup style is like the almtreegroup style, but has a set of links to the groups at the start of the glossary.

```
7911 \newglossarystyle{almtreehypergroup}{%
```

Base it on the glostylealmtree style:

```
7912 \setglossarystyle{almtree}%
```

Put the navigation links in the header

```
7913 \renewcommand*{\glossaryheader}{%  
7914 \par  
7915 \def\@gls@prevlevel{-1}%  
7916 \hangindent0pt\relax  
7917 \parindent0pt\relax  
7918 \textbf{\glsnavigation}\par\indexspace}%
```

Put a hypertarget at the start of each group

```
7919 \renewcommand*{\glsgroupheading}[1]{%  
7920 \par  
7921 \def\@gls@prevlevel{-1}%  
7922 \hangindent0pt\relax  
7923 \parindent0pt\relax  
7924 \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par  
7925 \indexspace}}
```

5 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.

```
7926 \NeedsTeXFormat{LaTeX2e}
7927 \ProvidesPackage{glossaries-compatible-207}[2011/04/02 v1.0 (NLCT)]
```

`\GlsAddXdyAttribute` Adds an attribute in old format.

```
7928 \ifglxsindy
7929   \renewcommand*\GlsAddXdyAttribute[1]{%
7930     \edef\@xdyattributes{\@xdyattributes ^^J \string"#1\string"}%
7931     \expandafter\toks@\expandafter{\@xdylocref}%
7932     \edef\@xdylocref{\the\toks@ ^^J%
7933       (markup-locref
7934       :open \string"\string~n\string\setentrycounter
7935         {\noexpand\glscounter}%
7936         \expandafter\string\csname#1\endcsname
7937         \expandafter\@gobble\string\{\string" ^^J
7938         :close \string"\expandafter\@gobble\string\}\string" ^^J
7939         :attr \string"#1\string"))}}
```

Only has an effect before `\writeist`:

```
7940 \fi
```

`\GlsAddXdyCounters`

```
7941 \renewcommand*\GlsAddXdyCounters[1]{%
7942   \GlossariesWarning{\string\GlsAddXdyCounters\space not available
7943     in compatibility mode.}%
7944 }
```

Add predefined attributes

```
7945 \GlsAddXdyAttribute{glslnumberformat}
7946 \GlsAddXdyAttribute{textrm}
7947 \GlsAddXdyAttribute{textsf}
7948 \GlsAddXdyAttribute{texttt}
7949 \GlsAddXdyAttribute{textbf}
7950 \GlsAddXdyAttribute{textmd}
7951 \GlsAddXdyAttribute{textit}
7952 \GlsAddXdyAttribute{textup}
7953 \GlsAddXdyAttribute{textsl}
7954 \GlsAddXdyAttribute{textsc}
7955 \GlsAddXdyAttribute{emph}
7956 \GlsAddXdyAttribute{glshypernumber}
7957 \GlsAddXdyAttribute{hyperrrm}
7958 \GlsAddXdyAttribute{hypersf}
7959 \GlsAddXdyAttribute{hypertt}
7960 \GlsAddXdyAttribute{hyperbf}
7961 \GlsAddXdyAttribute{hypermd}
```

```

7962 \GlsAddXdyAttribute{hyperit}
7963 \GlsAddXdyAttribute{hyperup}
7964 \GlsAddXdyAttribute{hypersl}
7965 \GlsAddXdyAttribute{hypersc}
7966 \GlsAddXdyAttribute{hyperemph}

```

\GlsAddXdyLocation Restore v2.07 definition:

```

7967 \ifglxsindy
7968   \renewcommand*{\GlsAddXdyLocation}[2]{%
7969     \edef\@xdyuserlocationdefs{%
7970       \@xdyuserlocationdefs ^^J%
7971       (define-location-class \string"#1\string"^^J\space\space
7972       \space(#2))
7973     }%
7974     \edef\@xdyuserlocationnames{%
7975       \@xdyuserlocationnames^^J\space\space\space
7976       \string"#1\string"}%
7977   }
7978 \fi

```

\@do@wrglossary

```

7979 \renewcommand{\@do@wrglossary}[1]{%

```

Determine whether to use xindy or makeindex syntax

```

7980 \ifglxsindy

```

Need to determine if the formatting information starts with a (or) indicating a range.

```

7981 \expandafter\@glo@check@mkidxrangechar\@glsnumberformat\@nil
7982 \def\@glo@range{}%
7983 \expandafter\if\@glo@prefix(\relax
7984   \def\@glo@range{:open-range}%
7985 \else
7986   \expandafter\if\@glo@prefix)\relax
7987   \def\@glo@range{:close-range}%
7988 \fi
7989 \fi

```

Get the location and escape any special characters

```

7990 \protected@edef\@glslocref{\theglentrycounter}%
7991 \@gls@checkmkidxchars\@glslocref

```

Write to the glossary file using xindy syntax.

```

7992 \glossary[\csname glo@#1@type\endcsname]{%
7993 (indexentry :tkey (\csname glo@#1@index\endcsname)
7994   :locref \string"\@glslocref\string" %
7995   :attr \string"\@glo@suffix\string" \@glo@range
7996 )
7997 }%
7998 \else

```

Convert the format information into the format required for makeindex

```
7999 \set@glo@numformat\@glo@numfmt\@gls@counter\@glsnumberformat
```

Write to the glossary file using makeindex syntax.

```
8000 \glossary[\csname glo@#1@type\endcsname]{%
8001 \string\glossaryentry{\csname glo@#1@index\endcsname
8002 \@gls@encapchar\@glo@numfmt}\@theglsentrycounter}}%
8003 \fi
8004 }
```

\set@glo@numformat Only had 3 arguments in v2.07

```
8005 \def\set@glo@numformat#1#2#3{%
8006 \expandafter\@glo@check@mkidxrangechar#3\@nil
8007 \protected@edef#1{%
8008 \@glo@prefix setentrycounter[] {#2}%
8009 \expandafter\string\csname\@glo@suffix\endcsname
8010 }%
8011 \@gls@checkmkidxchars#1%
8012 }
```

\writeist Redefine \writeist back to the way it was in v2.07, but change \istfile to \glswrite.

```
8013 \ifglxindy
8014 \def\writeist{%
8015 \openout\glswrite=\istfilename
8016 \write\glswrite{;; xindy style file created by the glossaries
8017 package in compatible-2.07 mode}%
8018 \write\glswrite{;; for document '\jobname' on
8019 \the\year-\the\month-\the\day}%
8020 \write\glswrite{^^J; required styles^^J}
8021 \@for\@xdystyle:=\@xdyrequiredstyles\do{%
8022 \ifx\@xdystyle\@empty
8023 \else
8024 \protected@write\glswrite{{(require
8025 \string"\@xdystyle.xdy\string")}}%
8026 \fi
8027 }%
8028 \write\glswrite{^^J%
8029 ; list of allowed attributes (number formats)^^J}%
8030 \write\glswrite{(define-attributes ((\@xdyattributes)))}%
8031 \write\glswrite{^^J; user defined alphabets^^J}%
8032 \write\glswrite{\@xdyuseralphabets}%
8033 \write\glswrite{^^J; location class definitions^^J}%
8034 \protected@edef\@gls@roman{\@roman{0}\string"
8035 \string"roman-numbers-lowercase\string" :sep \string"}}%
8036 \@onelevel@sanitize\@gls@roman
8037 \edef\@tmp{\string" \string"roman-numbers-lowercase\string"
8038 :sep \string"}%
8039 \@onelevel@sanitize\@tmp
```

```

8040 \ifx\@tmp\@gls@roman
8041 \write\glswrite{(define-location-class
8042 \string"roman-page-numbers\string"^^J\space\space\space
8043 (\string"roman-numbers-lowercase\string")
8044 :min-range-length \@glsminrange)}}%
8045 \else
8046 \write\glswrite{(define-location-class
8047 \string"roman-page-numbers\string"^^J\space\space\space
8048 (:sep "\@gls@roman")
8049 :min-range-length \@glsminrange)}}%
8050 \fi
8051 \write\glswrite{(define-location-class
8052 \string"Roman-page-numbers\string"^^J\space\space\space
8053 (\string"roman-numbers-uppercase\string")
8054 :min-range-length \@glsminrange)}}%
8055 \write\glswrite{(define-location-class
8056 \string"arabic-page-numbers\string"^^J\space\space\space
8057 (\string"arabic-numbers\string")
8058 :min-range-length \@glsminrange)}}%
8059 \write\glswrite{(define-location-class
8060 \string"alpha-page-numbers\string"^^J\space\space\space
8061 (\string"alpha\string")
8062 :min-range-length \@glsminrange)}}%
8063 \write\glswrite{(define-location-class
8064 \string"Alpha-page-numbers\string"^^J\space\space\space
8065 (\string"ALPHA\string")
8066 :min-range-length \@glsminrange)}}%
8067 \write\glswrite{(define-location-class
8068 \string"Appendix-page-numbers\string"^^J\space\space\space
8069 (\string"ALPHA\string"
8070 :sep \string"\@glsAlphacompositor\string"
8071 \string"arabic-numbers\string")
8072 :min-range-length \@glsminrange)}}%
8073 \write\glswrite{(define-location-class
8074 \string"arabic-section-numbers\string"^^J\space\space\space
8075 (\string"arabic-numbers\string"
8076 :sep \string"\glscompositor\string"
8077 \string"arabic-numbers\string")
8078 :min-range-length \@glsminrange)}}%
8079 \write\glswrite{^^J; user defined location classes}%
8080 \write\glswrite{\@xdyuserlocationdefs}%
8081 \write\glswrite{^^J; define cross-reference class^^J}%
8082 \write\glswrite{(define-crossref-class \string"see\string"
8083 :unverified )}%
8084 \write\glswrite{(markup-crossref-list
8085 :class \string"see\string"^^J\space\space\space
8086 :open \string"\string\glsseeformat\string"
8087 :close \string"{}\string")}%
8088 \write\glswrite{^^J; define the order of the location classes}%

```

```

8089 \write\glswrite{(define-location-class-order
8090   (\@xdylocationclassorder))}%
8091 \write\glswrite{^^J; define the glossary markup^^J}%
8092 \write\glswrite{(markup-index^^J\space\space\space
8093   :open \string"\string
8094   \glossarysection[\string\glossarytoctitle]{\string
8095   \glossarytitle}\string\glossarypreamble\string~n\string\begin
8096   {theglossary}\string\glossaryheader\string~n\string" ^^J\space
8097   \space\space:close \string"\expandafter\@gobble
8098   \string%\string~n\string
8099   \end{theglossary}\string\glossarypostamble
8100   \string~n\string" ^^J\space\space\space
8101   :tree)}}%
8102 \write\glswrite{(markup-letter-group-list
8103   :sep \string"\string\glsgroupskip\string~n\string"))}%
8104 \write\glswrite{(markup-indexentry
8105   :open \string"\string\relax \string\glresetentrylist
8106   \string~n\string"))}%
8107 \write\glswrite{(markup-locclass-list :open
8108   \string"\glsoopenbrace\string\glossaryentrynumbers
8109   \glsoopenbrace\string\relax\space \string"^^J\space\space\space
8110   :sep \string", \string"
8111   :close \string"\glsclosebrace\glsclosebrace\string"))}%
8112 \write\glswrite{(markup-locref-list
8113   :sep \string"\string\delimN\space\string"))}%
8114 \write\glswrite{(markup-range
8115   :sep \string"\string\delimR\space\string"))}%
8116 \@onelevel@sanitize\gls@suffixF
8117 \@onelevel@sanitize\gls@suffixFF
8118 \ifx\gls@suffixF\@empty
8119 \else
8120   \write\glswrite{(markup-range
8121     :close "\gls@suffixF" :length 1 :ignore-end)}}%
8122 \fi
8123 \ifx\gls@suffixFF\@empty
8124 \else
8125   \write\glswrite{(markup-range
8126     :close "\gls@suffixFF" :length 2 :ignore-end)}}%
8127 \fi
8128 \write\glswrite{^^J; define format to use for locations^^J}%
8129 \write\glswrite{\@xdylocref}%
8130 \write\glswrite{^^J; define letter group list format^^J}%
8131 \write\glswrite{(markup-letter-group-list
8132   :sep \string"\string\glsgroupskip\string~n\string"))}%
8133 \write\glswrite{^^J; letter group headings^^J}%
8134 \write\glswrite{(markup-letter-group
8135   :open-head \string"\string\glsgroupheading
8136   \glsoopenbrace\string"^^J\space\space\space
8137   :close-head \string"\glsclosebrace\string"))}%

```

```

8138 \write\glswrite{^^J; additional letter groups^^J}%
8139 \write\glswrite{\@xdylettergroups}%
8140 \write\glswrite{^^J; additional sort rules^^J}
8141 \write\glswrite{\@xdysortrules}%
8142 \noist}
8143 \else
8144 \edef\@gls@actualchar{\string?}
8145 \edef\@gls@encapchar{\string|}
8146 \edef\@gls@levelchar{\string!}
8147 \edef\@gls@quotechar{\string"}
8148 \def\writeist{\relax
8149 \openout\glswrite=\istfilename
8150 \write\glswrite{\expandafter\@gobble\string\% makeindex style file
8151 created by the glossaries package}
8152 \write\glswrite{\expandafter\@gobble\string\% for document
8153 '\jobname' on \the\year-\the\month-\the\day}
8154 \write\glswrite{actual '\@gls@actualchar'}
8155 \write\glswrite{encap '\@gls@encapchar'}
8156 \write\glswrite{level '\@gls@levelchar'}
8157 \write\glswrite{quote '\@gls@quotechar'}
8158 \write\glswrite{keyword \string"\string\glossaryentry\string"}
8159 \write\glswrite{preamble \string"\string\glossarysection[\string
8160 \glossarytoctitle]{\string\glossarytitle}\string
8161 \glossarypreamble\string\n\string\begin{theglossary}\string
8162 \glossaryheader\string\n\string"}
8163 \write\glswrite{postamble \string"\string%\string\n\string
8164 \end{theglossary}\string\glossarypostamble\string\n
8165 \string"}
8166 \write\glswrite{group_skip \string"\string\glsgroupskip\string\n
8167 \string"}
8168 \write\glswrite{item_0 \string"\string%\string\n\string"}
8169 \write\glswrite{item_1 \string"\string%\string\n\string"}
8170 \write\glswrite{item_2 \string"\string%\string\n\string"}
8171 \write\glswrite{item_01 \string"\string%\string\n\string"}
8172 \write\glswrite{item_x1
8173 \string"\string\relax \string\glsresetentrylist\string\n
8174 \string"}
8175 \write\glswrite{item_12 \string"\string%\string\n\string"}
8176 \write\glswrite{item_x2
8177 \string"\string\relax \string\glsresetentrylist\string\n
8178 \string"}
8179 \write\glswrite{delim_0 \string"\string\{\string
8180 \glossaryentrynumbers\string\{\string\relax \string"}
8181 \write\glswrite{delim_1 \string"\string\{\string
8182 \glossaryentrynumbers\string\{\string\relax \string"}
8183 \write\glswrite{delim_2 \string"\string\{\string
8184 \glossaryentrynumbers\string\{\string\relax \string"}
8185 \write\glswrite{delim_t \string"\string\}\string\}\string"}
8186 \write\glswrite{delim_n \string"\string\delimN \string"}

```

```

8187 \write\glswrite{delim_r \string"\string\\delimR \string"}
8188 \write\glswrite{headings_flag 1}
8189 \write\glswrite{heading_prefix
8190   \string"\string\\glsgroupheading\string\{\string"}
8191 \write\glswrite{heading_suffix
8192   \string"\string\\}\string\\relax
8193   \string\\glsresetentrylist \string"}
8194 \write\glswrite{symhead_positive \string"glssymbols\string"}
8195 \write\glswrite{numhead_positive \string"glssymbols\string"}
8196 \write\glswrite{page_compositor \string"\glscompositor\string"}
8197 \@gls@escbsdq\gls@suffixF
8198 \@gls@escbsdq\gls@suffixFF
8199 \ifx\gls@suffixF\@empty
8200 \else
8201   \write\glswrite{suffix_2p \string"\gls@suffixF\string"}
8202 \fi
8203 \ifx\gls@suffixFF\@empty
8204 \else
8205   \write\glswrite{suffix_3p \string"\gls@suffixFF\string"}
8206 \fi
8207 \noist
8208 }
8209 \fi

```

\noist

```
8210 \renewcommand*{\noist}{\let\writeist\relax}
```

Compatibility macros.

```

8211 \NeedsTeXFormat{LaTeX2e}
8212 \ProvidesPackage{glossaries-compatible-307}[2013/11/14 v4.0 (NLCT)]

```

Compatibility macros for predefined glossary styles:

compatglossarystyle Defines a compatibility glossary style.

```

8213 \newcommand{\compatglossarystyle}[2]{%
8214   \ifcsundef{@glscompstyle@#1}%
8215   {%
8216     \csdef{@glscompstyle@#1}{#2}%
8217   }%
8218   {%
8219     \PackageError{glossaries}{Glossary compatibility style ‘#1’ is already defined}{}%
8220   }%
8221 }

```

Backward compatible inline style.

```

8222 \compatglossarystyle{inline}{%
8223   \renewcommand{\glossaryentryfield}[5]{%
8224     \glsinlinedopostchild
8225     \gls@inlinesep
8226     \def\glo@desc{##3}%

```

```

8227 \def\@no@post@desc{\nopostdesc}%
8228 \glsentryitem{##1}\glsinlinenameformat{##1}{##2}%
8229 \ifx\glo@desc\@no@post@desc
8230 \glsinlineemptydescformat{##4}{##5}%
8231 \else
8232 \ifstrempy{##3}%
8233 {\glsinlineemptydescformat{##4}{##5}}%
8234 {\glsinlinedescformat{##3}{##4}{##5}}%
8235 \fi
8236 \ifglshaschildren{##1}%
8237 {%
8238 \glsresetsubentrycounter
8239 \glsinlineparentchildseparator
8240 \def\gls@inlinesubsep{}%
8241 \def\gls@inlinepostchild{\glsinlinepostchild}%
8242 }%
8243 {}%
8244 \def\gls@inlinesep{\glsinlineseparator}%
8245 }%

```

Sub-entries display description:

```

8246 \renewcommand{\glossarysubentryfield}[6]{%
8247 \gls@inlinesubsep%
8248 \glsinlinesubnameformat{##2}{##3}%
8249 \glssubentryitem{##2}\glsinlinesubdescformat{##4}{##5}{##6}%
8250 \def\gls@inlinesubsep{\glsinlinesubseparator}%
8251 }%
8252 }

```

Backward compatible list style.

```

8253 \compatglossarystyle{list}{%
8254 \renewcommand*\glossaryentryfield[5]{%
8255 \item[\glsentryitem{##1}\glstarget{##1}{##2}]
8256 ##3\glspostdescription\space ##5}%

```

Sub-entries continue on the same line:

```

8257 \renewcommand*\glossarysubentryfield[6]{%
8258 \glssubentryitem{##2}%
8259 \glstarget{##2}{\strut}##4\glspostdescription\space ##6.}%
8260 }

```

Backward compatible listgroup style.

```

8261 \compatglossarystyle{listgroup}{%
8262 \csuse{@glscompstyle@list}%
8263 }%

```

Backward compatible listhypergroup style.

```

8264 \compatglossarystyle{listhypergroup}{%
8265 \csuse{@glscompstyle@list}%
8266 }%

```

Backward compatible altlist style.

```

8267 \compatglossarystyle{altlist}{%
8268   \renewcommand*{\glossaryentryfield}[5]{%
8269     \item[\glentryitem{##1}\glstarget{##1}{##2}]%
8270     \mbox{}\par\nobreak\@afterheading
8271     ##3\glspostdescription\space ##5}%
8272   \renewcommand{\glossarysubentryfield}[6]{%
8273     \par
8274     \glssubentryitem{##2}%
8275     \glstarget{##2}{\strut}##4\glspostdescription\space ##6}%
8276 }%

```

Backward compatible altlistgroup style.

```

8277 \compatglossarystyle{altlistgroup}{%
8278   \csuse{@glscmpstyle@altlist}%
8279 }%

```

Backward compatible altlisthypergroup style.

```

8280 \compatglossarystyle{altlisthypergroup}{%
8281   \csuse{@glscmpstyle@altlist}%
8282 }%

```

Backward compatible listdotted style.

```

8283 \compatglossarystyle{listdotted}{%
8284   \renewcommand*{\glossaryentryfield}[5]{%
8285     \item[\makebox[\glslistdottedwidth][l]{%
8286       \glentryitem{##1}\glstarget{##1}{##2}%
8287       \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##3}%
8288   \renewcommand*{\glossarysubentryfield}[6]{%
8289     \item[\makebox[\glslistdottedwidth][l]{%
8290       \glssubentryitem{##2}%
8291       \glstarget{##2}{##3}%
8292       \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##4}%
8293 }%

```

Backward compatible sublistdotted style.

```

8294 \compatglossarystyle{sublistdotted}{%
8295   \csuse{@glscmpstyle@listdotted}%
8296   \renewcommand*{\glossaryentryfield}[5]{%
8297     \item[\glentryitem{##1}\glstarget{##1}{##2}]}%
8298 }%

```

Backward compatible long style.

```

8299 \compatglossarystyle{long}{%
8300   \renewcommand*{\glossaryentryfield}[5]{%
8301     \glentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\}%
8302   \renewcommand*{\glossarysubentryfield}[6]{%
8303     &
8304     \glssubentryitem{##2}%
8305     \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\}%
8306 }%

```

Backward compatible longborder style.

```

8307 \compatglossarystyle{longborder}{%
8308  \csuse{@glscompstyle@long}%
8309 }%

```

Backward compatible longheader style.

```

8310 \compatglossarystyle{longheader}{%
8311  \csuse{@glscompstyle@long}%
8312 }%

```

Backward compatible longheaderborder style.

```

8313 \compatglossarystyle{longheaderborder}{%
8314  \csuse{@glscompstyle@long}%
8315 }%

```

Backward compatible long3col style.

```

8316 \compatglossarystyle{long3col}{%
8317  \renewcommand*{\glossaryentryfield}[5]{%
8318    \glstryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\\}%
8319  \renewcommand*{\glossarysubentryfield}[6]{%
8320    &
8321    \glssubentryitem{##2}%
8322    \glstarget{##2}{\strut}##4 & ##6\\}%
8323 }%

```

Backward compatible long3colborder style.

```

8324 \compatglossarystyle{long3colborder}{%
8325  \csuse{@glscompstyle@long3col}%
8326 }%

```

Backward compatible long3colheader style.

```

8327 \compatglossarystyle{long3colheader}{%
8328  \csuse{@glscompstyle@long3col}%
8329 }%

```

Backward compatible long3colheaderborder style.

```

8330 \compatglossarystyle{long3colheaderborder}{%
8331  \csuse{@glscompstyle@long3col}%
8332 }%

```

Backward compatible long4col style.

```

8333 \compatglossarystyle{long4col}{%
8334  \renewcommand*{\glossaryentryfield}[5]{%
8335    \glstryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\\}%
8336  \renewcommand*{\glossarysubentryfield}[6]{%
8337    &
8338    \glssubentryitem{##2}%
8339    \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%
8340 }%

```

Backward compatible long4colheader style.

```

8341 \compatglossarystyle{long4colheader}{%
8342  \csuse{@glscompstyle@long4col}%
8343 }%

```

Backward compatible long4colborder style.

```
8344 \compatglossarystyle{long4colborder}{%  
8345 \csuse{@glscompstyle@long4col}%  
8346 }%
```

Backward compatible long4colheaderborder style.

```
8347 \compatglossarystyle{long4colheaderborder}{%  
8348 \csuse{@glscompstyle@long4col}%  
8349 }%
```

Backward compatible altlong4col style.

```
8350 \compatglossarystyle{altlong4col}{%  
8351 \csuse{@glscompstyle@long4col}%  
8352 }%
```

Backward compatible altlong4colheader style.

```
8353 \compatglossarystyle{altlong4colheader}{%  
8354 \csuse{@glscompstyle@long4col}%  
8355 }%
```

Backward compatible altlong4colborder style.

```
8356 \compatglossarystyle{altlong4colborder}{%  
8357 \csuse{@glscompstyle@long4col}%  
8358 }%
```

Backward compatible altlong4colheaderborder style.

```
8359 \compatglossarystyle{altlong4colheaderborder}{%  
8360 \csuse{@glscompstyle@long4col}%  
8361 }%
```

Backward compatible long style.

```
8362 \compatglossarystyle{longragged}{%  
8363 \renewcommand*{\glossaryentryfield}[5]{%  
8364 \glsentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%  
8365 \tabularnewline}%  
8366 \renewcommand*{\glossarysubentryfield}[6]{%  
8367 &  
8368 \glssubentryitem{##2}%  
8369 \glstarget{##2}{\strut}##4\glspostdescription\space ##6%  
8370 \tabularnewline}%  
8371 }%
```

Backward compatible longraggedborder style.

```
8372 \compatglossarystyle{longraggedborder}{%  
8373 \csuse{@glscompstyle@longragged}%  
8374 }%
```

Backward compatible longraggedheader style.

```
8375 \compatglossarystyle{longraggedheader}{%  
8376 \csuse{@glscompstyle@longragged}%  
8377 }%
```

Backward compatible longraggedheaderborder style.

```
8378 \compatglossarystyle{longraggedheaderborder}{%
8379 \csuse{@glscmpstyle@longragged}%
8380 }%
```

Backward compatible longragged3col style.

```
8381 \compatglossarystyle{longragged3col}{%
8382 \renewcommand*{\glossaryentryfield}[5]{%
8383 \glentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\tabularnewline}%
8384 \renewcommand*{\glossarysubentryfield}[6]{%
8385 &
8386 \glssubentryitem{##2}%
8387 \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%
8388 }%
```

Backward compatible longragged3colborder style.

```
8389 \compatglossarystyle{longragged3colborder}{%
8390 \csuse{@glscmpstyle@longragged3col}%
8391 }%
```

Backward compatible longragged3colheader style.

```
8392 \compatglossarystyle{longragged3colheader}{%
8393 \csuse{@glscmpstyle@longragged3col}%
8394 }%
```

Backward compatible longragged3colheaderborder style.

```
8395 \compatglossarystyle{longragged3colheaderborder}{%
8396 \csuse{@glscmpstyle@longragged3col}%
8397 }%
```

Backward compatible altlongragged4col style.

```
8398 \compatglossarystyle{altlongragged4col}{%
8399 \renewcommand*{\glossaryentryfield}[5]{%
8400 \glentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\tabularnewline}%
8401 \renewcommand*{\glossarysubentryfield}[6]{%
8402 &
8403 \glssubentryitem{##2}%
8404 \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%
8405 }%
```

Backward compatible altlongragged4colheader style.

```
8406 \compatglossarystyle{altlongragged4colheader}{%
8407 \csuse{@glscmpstyle@altlong4col}%
8408 }%
```

Backward compatible altlongragged4colborder style.

```
8409 \compatglossarystyle{altlongragged4colborder}{%
8410 \csuse{@glscmpstyle@altlong4col}%
8411 }%
```

Backward compatible altlongragged4colheaderborder style.

```
8412 \compatglossarystyle{altlongragged4colheaderborder}{%
```

```

8413 \csuse{@glscompstyle@altlong4col}%
8414 }%

```

Backward compatible index style.

```

8415 \compatglossarystyle{index}{%
8416   \renewcommand*{\glossaryentryfield}[5]{%
8417     \item\glentryitem{##1}\textbf{\glstarget{##1}{##2}}%
8418     \ifx\relax##4\relax
8419       \else
8420         \space{##4}%
8421       \fi
8422     \space ##3\glspostdescription \space ##5}%
8423   \renewcommand*{\glossarysubentryfield}[6]{%
8424     \ifcase##1\relax
8425       % level 0
8426       \item
8427     \or
8428       % level 1
8429       \subitem
8430       \glssubentryitem{##2}%
8431     \else
8432       % all other levels
8433       \subsubitem
8434     \fi
8435     \textbf{\glstarget{##2}{##3}}%
8436     \ifx\relax##5\relax
8437     \else
8438       \space{##5}%
8439     \fi
8440     \space##4\glspostdescription\space ##6}%
8441 }%

```

Backward compatible indexgroup style.

```

8442 \compatglossarystyle{indexgroup}{%
8443   \csuse{@glscompstyle@index}%
8444 }%

```

Backward compatible indexhypergroup style.

```

8445 \compatglossarystyle{indexhypergroup}{%
8446   \csuse{@glscompstyle@index}%
8447 }%

```

Backward compatible tree style.

```

8448 \compatglossarystyle{tree}{%
8449   \renewcommand{\glossaryentryfield}[5]{%
8450     \hangindent0pt\relax
8451     \parindent0pt\relax
8452     \glentryitem{##1}\textbf{\glstarget{##1}{##2}}%
8453     \ifx\relax##4\relax
8454     \else
8455       \space{##4}%

```

```

8456 \fi
8457 \space ##3\glspostdescription \space ##5\par}%
8458 \renewcommand{\glossarysubentryfield}[6]{%
8459 \hangindent##1\glstreeindent\relax
8460 \parindent##1\glstreeindent\relax
8461 \ifnum##1=1\relax
8462 \glssubentryitem{##2}%
8463 \fi
8464 \textbf{\glstarget{##2}{##3}}%
8465 \ifx\relax##5\relax
8466 \else
8467 \space(##5)%
8468 \fi
8469 \space##4\glspostdescription\space ##6\par}%
8470 }%

```

Backward compatible treegroup style.

```

8471 \compatglossarystyle{treegroup}{%
8472 \csuse{@glscmpstyle@tree}%
8473 }%

```

Backward compatible treehypergroup style.

```

8474 \compatglossarystyle{treehypergroup}{%
8475 \csuse{@glscmpstyle@tree}%
8476 }%

```

Backward compatible treenoname style.

```

8477 \compatglossarystyle{treenoname}{%
8478 \renewcommand{\glossaryentryfield}[5]{%
8479 \hangindent0pt\relax
8480 \parindent0pt\relax
8481 \glstryitem{##1}\textbf{\glstarget{##1}{##2}}%
8482 \ifx\relax##4\relax
8483 \else
8484 \space(##4)%
8485 \fi
8486 \space ##3\glspostdescription \space ##5\par}%
8487 \renewcommand{\glossarysubentryfield}[6]{%
8488 \hangindent##1\glstreeindent\relax
8489 \parindent##1\glstreeindent\relax
8490 \ifnum##1=1\relax
8491 \glssubentryitem{##2}%
8492 \fi
8493 \glstarget{##2}{\strut}%
8494 ##4\glspostdescription\space ##6\par}%
8495 }%

```

Backward compatible treenonamegroup style.

```

8496 \compatglossarystyle{treenonamegroup}{%
8497 \csuse{@glscmpstyle@treenoname}%
8498 }%

```

Backward compatible treenonamehypergroup style.

```
8499 \compatglossarystyle{treenonamehypergroup}{%
8500 \csuse{@glscompstyle@treenoname}%
8501 }%
```

Backward compatible alttree style.

```
8502 \compatglossarystyle{alttree}{%
8503 \renewcommand{\glossaryentryfield}[5]{%
8504 \ifnum\@gls@prevlevel=0\relax
8505 \else
8506 \settowidth{\glstreeindent}{\textbf{\@glswidestname\space}}%
8507 \hangindent\glstreeindent
8508 \parindent\glstreeindent
8509 \fi
8510 \makebox[0pt][r]{\makebox[\glstreeindent][l]{%
8511 \glstryitem{##1}\textbf{\glstarget{##1}{##2}}}%
8512 \ifx\relax##4\relax
8513 \else
8514 (##4)\space
8515 \fi
8516 ##3\glspostdescription \space ##5\par
8517 \def\@gls@prevlevel{0}%
8518 }%
8519 \renewcommand{\glossarysubentryfield}[6]{%
8520 \ifnum##1=1\relax
8521 \glssubentryitem{##2}%
8522 \fi
8523 \ifnum\@gls@prevlevel=##1\relax
8524 \else
8525 \@ifundefined{@glswidestname\romannumeral##1}{%
8526 \settowidth{\gls@tmplen}{\textbf{\@glswidestname\space}}}%
8527 \settowidth{\gls@tmplen}{\textbf{%
8528 \csname @glswidestname\romannumeral##1\endcsname\space}}}%
8529 \ifnum\@gls@prevlevel<##1\relax
8530 \setlength\glstreeindent\gls@tmplen
8531 \addtolength\glstreeindent\parindent
8532 \parindent\glstreeindent
8533 \else
8534 \@ifundefined{@glswidestname\romannumeral\@gls@prevlevel}{%
8535 \settowidth{\glstreeindent}{\textbf{%
8536 \@glswidestname\space}}}%
8537 \settowidth{\glstreeindent}{\textbf{%
8538 \csname @glswidestname\romannumeral\@gls@prevlevel
8539 \endcsname\space}}}%
8540 \addtolength\parindent{-\glstreeindent}%
8541 \setlength\glstreeindent\parindent
8542 \fi
8543 \fi
8544 \hangindent\glstreeindent
8545 \makebox[0pt][r]{\makebox[\gls@tmplen][l]{%
```

```

8546      \textbf{\glstarget{##2}{##3}}}%
8547      \ifx##5\relax\relax
8548      \else
8549          (##5)\space
8550      \fi
8551      ##4\glspostdescription\space ##6\par
8552      \def\@gls@prevlevel{##1}%
8553  }%
8554 }%

```

Backward compatible alttreegroup style.

```

8555 \compatglossarystyle{alttreegroup}{%
8556 \csuse{@glscompstyle@almtree}%
8557 }%

```

Backward compatible alttreehypergroup style.

```

8558 \compatglossarystyle{alttreehypergroup}{%
8559 \csuse{@glscompstyle@almtree}%
8560 }%

```

Backward compatible mcolindex style.

```

8561 \compatglossarystyle{mcolindex}{%
8562 \csuse{@glscompstyle@index}%
8563 }%

```

Backward compatible mcolindexgroup style.

```

8564 \compatglossarystyle{mcolindexgroup}{%
8565 \csuse{@glscompstyle@index}%
8566 }%

```

Backward compatible mcolindexhypergroup style.

```

8567 \compatglossarystyle{mcolindexhypergroup}{%
8568 \csuse{@glscompstyle@index}%
8569 }%

```

Backward compatible mcoltree style.

```

8570 \compatglossarystyle{mcoltree}{%
8571 \csuse{@glscompstyle@tree}%
8572 }%

```

Backward compatible mcoltreegroup style.

```

8573 \compatglossarystyle{mcolindextreegroup}{%
8574 \csuse{@glscompstyle@tree}%
8575 }%

```

Backward compatible mcoltreehypergroup style.

```

8576 \compatglossarystyle{mcolindextreehypergroup}{%
8577 \csuse{@glscompstyle@tree}%
8578 }%

```

Backward compatible mcoltreenoname style.

```

8579 \compatglossarystyle{mcoltreenoname}{%
8580 \csuse{@glscompstyle@tree}%
8581 }%

```

Backward compatible mcoltreenonamegroup style.

```
8582 \compatglossarystyle{mcoltreenonamegroup}{%  
8583 \csuse{@glscompstyle@tree}%  
8584 }%
```

Backward compatible mcoltreenonamehypergroup style.

```
8585 \compatglossarystyle{mcoltreenonamehypergroup}{%  
8586 \csuse{@glscompstyle@tree}%  
8587 }%
```

Backward compatible mcolalmtree style.

```
8588 \compatglossarystyle{mcolalmtree}{%  
8589 \csuse{@glscompstyle@almtree}%  
8590 }%
```

Backward compatible mcolalmtreegroup style.

```
8591 \compatglossarystyle{mcolalmtreegroup}{%  
8592 \csuse{@glscompstyle@almtree}%  
8593 }%
```

Backward compatible mcolalmtreehypergroup style.

```
8594 \compatglossarystyle{mcolalmtreehypergroup}{%  
8595 \csuse{@glscompstyle@almtree}%  
8596 }%
```

Backward compatible superragged style.

```
8597 \compatglossarystyle{superragged}{%  
8598 \renewcommand*{\glossaryentryfield}[5]{%  
8599 \glstryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%  
8600 \tabularnewline}%  
8601 \renewcommand*{\glossarysubentryfield}[6]{%  
8602 &  
8603 \glssubentryitem{##2}%  
8604 \glstarget{##2}{\strut}##4\glspostdescription\space ##6%  
8605 \tabularnewline}%  
8606 }%
```

Backward compatible superraggedborder style.

```
8607 \compatglossarystyle{superraggedborder}{%  
8608 \csuse{@glscompstyle@superragged}%  
8609 }%
```

Backward compatible superraggedheader style.

```
8610 \compatglossarystyle{superraggedheader}{%  
8611 \csuse{@glscompstyle@superragged}%  
8612 }%
```

Backward compatible superraggedheaderborder style.

```
8613 \compatglossarystyle{superraggedheaderborder}{%  
8614 \csuse{@glscompstyle@superragged}%  
8615 }%
```

Backward compatible superragged3col style.

```
8616 \compatglossarystyle{superragged3col}{%
8617   \renewcommand*{\glossaryentryfield}[5]{%
8618     \glstryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\tabularnewline}%
8619   \renewcommand*{\glossarysubentryfield}[6]{%
8620     &
8621     \glssubentryitem{##2}%
8622     \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%
8623 }%
```

Backward compatible superragged3colborder style.

```
8624 \compatglossarystyle{superragged3colborder}{%
8625   \csuse{@glscmpstyle@superragged3col}%
8626 }%
```

Backward compatible superragged3colheader style.

```
8627 \compatglossarystyle{superragged3colheader}{%
8628   \csuse{@glscmpstyle@superragged3col}%
8629 }%
```

Backward compatible superragged3colheaderborder style.

```
8630 \compatglossarystyle{superragged3colheaderborder}{%
8631   \csuse{@glscmpstyle@superragged3col}%
8632 }%
```

Backward compatible altsuperragged4col style.

```
8633 \compatglossarystyle{altsuperragged4col}{%
8634   \renewcommand*{\glossaryentryfield}[5]{%
8635     \glstryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\tabularnewline}%
8636   \renewcommand*{\glossarysubentryfield}[6]{%
8637     &
8638     \glssubentryitem{##2}%
8639     \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%
8640 }%
```

Backward compatible altsuperragged4colheader style.

```
8641 \compatglossarystyle{altsuperragged4colheader}{%
8642   \csuse{@glscmpstyle@altsuperragged4col}%
8643 }%
```

Backward compatible altsuperragged4colborder style.

```
8644 \compatglossarystyle{altsuperragged4colborder}{%
8645   \csuse{@glscmpstyle@altsuperragged4col}%
8646 }%
```

Backward compatible altsuperragged4colheaderborder style.

```
8647 \compatglossarystyle{altsuperragged4colheaderborder}{%
8648   \csuse{@glscmpstyle@altsuperragged4col}%
8649 }%
```

Backward compatible super style.

```
8650 \compatglossarystyle{super}{%
```

```

8651 \renewcommand*{\glossaryentryfield}[5]{%
8652   \glentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\}%
8653 \renewcommand*{\glossarysubentryfield}[6]{%
8654   &
8655   \glssubentryitem{##2}%
8656   \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\}%
8657 }%

```

Backward compatible superborder style.

```

8658 \compatglossarystyle{superborder}{%
8659 \csuse{@glscmpstyle@super}%
8660 }%

```

Backward compatible superheader style.

```

8661 \compatglossarystyle{superheader}{%
8662 \csuse{@glscmpstyle@super}%
8663 }%

```

Backward compatible superheaderborder style.

```

8664 \compatglossarystyle{superheaderborder}{%
8665 \csuse{@glscmpstyle@super}%
8666 }%

```

Backward compatible super3col style.

```

8667 \compatglossarystyle{super3col}{%
8668 \renewcommand*{\glossaryentryfield}[5]{%
8669   \glentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\\}%
8670 \renewcommand*{\glossarysubentryfield}[6]{%
8671   &
8672   \glssubentryitem{##2}%
8673   \glstarget{##2}{\strut}##4 & ##6\\}%
8674 }%

```

Backward compatible super3colborder style.

```

8675 \compatglossarystyle{super3colborder}{%
8676 \csuse{@glscmpstyle@super3col}%
8677 }%

```

Backward compatible super3colheader style.

```

8678 \compatglossarystyle{super3colheader}{%
8679 \csuse{@glscmpstyle@super3col}%
8680 }%

```

Backward compatible super3colheaderborder style.

```

8681 \compatglossarystyle{super3colheaderborder}{%
8682 \csuse{@glscmpstyle@super3col}%
8683 }%

```

Backward compatible super4col style.

```

8684 \compatglossarystyle{super4col}{%
8685 \renewcommand*{\glossaryentryfield}[5]{%
8686   \glentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\\}%

```

```

8687 \renewcommand*{\glossarysubentryfield}[6]{%
8688     &
8689     \glssubentryitem{##2}%
8690     \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%
8691 }%

```

Backward compatible super4colheader style.

```

8692 \compatglossarystyle{super4colheader}{%
8693 \csuse{@glscmpstyle@super4col}%
8694 }%

```

Backward compatible super4colborder style.

```

8695 \compatglossarystyle{super4colborder}{%
8696 \csuse{@glscmpstyle@super4col}%
8697 }%

```

Backward compatible super4colheaderborder style.

```

8698 \compatglossarystyle{super4colheaderborder}{%
8699 \csuse{@glscmpstyle@super4col}%
8700 }%

```

Backward compatible altsuper4col style.

```

8701 \compatglossarystyle{altsuper4col}{%
8702 \csuse{@glscmpstyle@super4col}%
8703 }%

```

Backward compatible altsuper4colheader style.

```

8704 \compatglossarystyle{altsuper4colheader}{%
8705 \csuse{@glscmpstyle@super4col}%
8706 }%

```

Backward compatible altsuper4colborder style.

```

8707 \compatglossarystyle{altsuper4colborder}{%
8708 \csuse{@glscmpstyle@super4col}%
8709 }%

```

Backward compatible altsuper4colheaderborder style.

```

8710 \compatglossarystyle{altsuper4colheaderborder}{%
8711 \csuse{@glscmpstyle@super4col}%
8712 }%

```

6 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.

```

8713 \NeedsTeXFormat{LaTeX2e}

```

Package version number now in line with main glossaries package number but will only be updated when glossaries-accsupp.sty is modified.

```

8714 \ProvidesPackage{glossaries-accsupp}[2013/11/14 v4.0 (NLCT)]
8715 Experimental glossaries accessibility]

```

Pass all options to glossaries:

```
8716 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{glossaries}}
```

Process options:

```
8717 \ProcessOptions
```

Override style compatibility macros:

```
8718 \newcommand*{\compatibleglossentry}[2]{%
8719   \toks@{#2}%
8720   \protected@edef\@do@glossentry{%
8721     \noexpand\accsuppglossaryentryfield{#1}%
8722     {\noexpand\glsnamefont
8723       {\expandafter\expandonce\csname glo@#1@name\endcsname}}}%
8724     {\expandafter\expandonce\csname glo@#1@desc\endcsname}%
8725     {\expandafter\expandonce\csname glo@#1@symbol\endcsname}%
8726     {\the\toks@}%
8727   }%
8728   \@do@glossentry
8729 }

8730 \newcommand*{\compatiblesubglossentry}[3]{%
8731   \toks@{#3}%
8732   \protected@edef\@do@subglossentry{%
8733     \noexpand\accsuppglossarysubentryfield{\number#1}%
8734     {#2}%
8735     {\noexpand\glsnamefont
8736       {\expandafter\expandonce\csname glo@#2@name\endcsname}}}%
8737     {\expandafter\expandonce\csname glo@#2@desc\endcsname}%
8738     {\expandafter\expandonce\csname glo@#2@symbol\endcsname}%
8739     {\the\toks@}%
8740   }%
8741   \@do@subglossentry
8742 }
```

Required packages:

```
8743 \RequirePackage{glossaries}
8744 \RequirePackage{accsupp}
```

6.1 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:

```
8745 \define@key{glossentry}{access}{%
8746   \def\@glo@access{#1}%
8747 }
```

textaccess The replacement text corresponding to the text key:

```
8748 \define@key{glossentry}{textaccess}{%
8749   \def\@glo@textaccess{#1}%
8750 }
```

firstaccess The replacement text corresponding to the first key:

```
8751 \define@key{glossentry}{firstaccess}{%
8752   \def\@glo@firstaccess{#1}%
8753 }
```

pluralaccess The replacement text corresponding to the plural key:

```
8754 \define@key{glossentry}{pluralaccess}{%
8755   \def\@glo@pluralaccess{#1}%
8756 }
```

firstpluralaccess The replacement text corresponding to the firstplural key:

```
8757 \define@key{glossentry}{firstpluralaccess}{%
8758   \def\@glo@firstpluralaccess{#1}%
8759 }
```

symbolaccess The replacement text corresponding to the symbol key:

```
8760 \define@key{glossentry}{symbolaccess}{%
8761   \def\@glo@symbolaccess{#1}%
8762 }
```

symbolpluralaccess The replacement text corresponding to the symbolplural key:

```
8763 \define@key{glossentry}{symbolpluralaccess}{%
8764   \def\@glo@symbolpluralaccess{#1}%
8765 }
```

descriptionaccess The replacement text corresponding to the description key:

```
8766 \define@key{glossentry}{descriptionaccess}{%
8767   \def\@glo@descaccess{#1}%
8768 }
```

descriptionpluralaccess The replacement text corresponding to the descriptionplural key:

```
8769 \define@key{glossentry}{descriptionpluralaccess}{%
8770   \def\@glo@descpluralaccess{#1}%
8771 }
```

shortaccess The replacement text corresponding to the short key:

```
8772 \define@key{glossentry}{shortaccess}{%
8773   \def\@glo@shortaccess{#1}%
8774 }
```

shortpluralaccess The replacement text corresponding to the shortplural key:

```
8775 \define@key{glossentry}{shortpluralaccess}{%
8776   \def\@glo@shortpluralaccess{#1}%
8777 }
```

longaccess The replacement text corresponding to the long key:

```
8778 \define@key{glossentry}{longaccess}{%
8779   \def\@glo@longaccess{#1}%
8780 }
```

longpluralaccess The replacement text corresponding to the longplural key:

```
8781 \define@key{glossentry}{longpluralaccess}{%
8782   \def\@glo@longpluralaccess{#1}%
8783 }
```

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={\glsaccsupp{inches}{in}}.

Append these new keys to \@gls@keymap:

```
8784 \appto\@gls@keymap{,%
8785   {access}{access},%
8786   {textaccess}{textaccess},%
8787   {firstaccess}{firstaccess},%
8788   {pluralaccess}{pluralaccess},%
8789   {firstpluralaccess}{firstpluralaccess},%
8790   {symbolaccess}{symbolaccess},%
8791   {symbolpluralaccess}{symbolpluralaccess},%
8792   {descaccess}{descaccess},%
8793   {descpluralaccess}{descpluralaccess},%
8794   {shortaccess}{shortaccess},%
8795   {shortpluralaccess}{shortpluralaccess},%
8796   {longaccess}{longaccess},%
8797   {longpluralaccess}{longpluralaccess}%
8798 }
```

\@gls@noaccess Indicates that no replacement text has been provided.

```
8799 \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):

```
8800 \let\@gls@oldnewglossaryentryprehook\@newglossaryentryprehook
8801 \renewcommand*{\@newglossaryentryprehook}{%
8802   \@gls@oldnewglossaryentryprehook
8803   \def\@glo@access{\@glo@symbol}%

```

Initialise the other keys:

```
8804   \def\@glo@textaccess{\@glo@access}%
8805   \def\@glo@firstaccess{\@glo@access}%
8806   \def\@glo@pluralaccess{\@glo@textaccess}%
8807   \def\@glo@firstpluralaccess{\@glo@pluralaccess}%
8808   \def\@glo@symbolaccess{\relax}%
8809   \def\@glo@symbolpluralaccess{\@glo@symbolaccess}%
8810   \def\@glo@descaccess{\relax}%
8811   \def\@glo@descpluralaccess{\@glo@descaccess}%

```

```

8812 \def\@glo@shortaccess{\relax}%
8813 \def\@glo@shortpluralaccess{\@glo@shortaccess}%
8814 \def\@glo@longaccess{\relax}%
8815 \def\@glo@longpluralaccess{\@glo@longaccess}%
8816 }

```

Add to the end hook:

```

8817 \let\@gls@oldnewglossaryentryposthook\@newglossaryentryposthook
8818 \renewcommand*{\@newglossaryentryposthook}{%
8819 \@gls@oldnewglossaryentryposthook

```

Store the access information:

```

8820 \expandafter
8821 \protected@xdef\csname glo@\@glo@label @access\endcsname{%
8822 \@glo@access}%
8823 \expandafter
8824 \protected@xdef\csname glo@\@glo@label @textaccess\endcsname{%
8825 \@glo@textaccess}%
8826 \expandafter
8827 \protected@xdef\csname glo@\@glo@label @firstaccess\endcsname{%
8828 \@glo@firstaccess}%
8829 \expandafter
8830 \protected@xdef\csname glo@\@glo@label @pluralaccess\endcsname{%
8831 \@glo@pluralaccess}%
8832 \expandafter
8833 \protected@xdef\csname glo@\@glo@label @firstpluralaccess\endcsname{%
8834 \@glo@firstpluralaccess}%
8835 \expandafter
8836 \protected@xdef\csname glo@\@glo@label @symbolaccess\endcsname{%
8837 \@glo@symbolaccess}%
8838 \expandafter
8839 \protected@xdef\csname glo@\@glo@label @symbolpluralaccess\endcsname{%
8840 \@glo@symbolpluralaccess}%
8841 \expandafter
8842 \protected@xdef\csname glo@\@glo@label @descaccess\endcsname{%
8843 \@glo@descaccess}%
8844 \expandafter
8845 \protected@xdef\csname glo@\@glo@label @descpluralaccess\endcsname{%
8846 \@glo@descpluralaccess}%
8847 \expandafter
8848 \protected@xdef\csname glo@\@glo@label @shortaccess\endcsname{%
8849 \@glo@shortaccess}%
8850 \expandafter
8851 \protected@xdef\csname glo@\@glo@label @shortpluralaccess\endcsname{%
8852 \@glo@shortpluralaccess}%
8853 \expandafter
8854 \protected@xdef\csname glo@\@glo@label @longaccess\endcsname{%
8855 \@glo@longaccess}%
8856 \expandafter
8857 \protected@xdef\csname glo@\@glo@label @longpluralaccess\endcsname{%

```

```

8858      \@glo@longpluralaccess}%
8859 }

```

6.2 Accessing Replacement Text

`\glsentryaccess` Get the value of the access key for the entry with the given label:

```

8860 \newcommand*{\glsentryaccess}[1]{%
8861   \csname glo@#1@access\endcsname
8862 }

```

`\glsentrytextaccess` Get the value of the textaccess key for the entry with the given label:

```

8863 \newcommand*{\glsentrytextaccess}[1]{%
8864   \csname glo@#1@textaccess\endcsname
8865 }

```

`\glsentryfirstaccess` Get the value of the firstaccess key for the entry with the given label:

```

8866 \newcommand*{\glsentryfirstaccess}[1]{%
8867   \csname glo@#1@firstaccess\endcsname
8868 }

```

`\glsentrypluralaccess` Get the value of the pluralaccess key for the entry with the given label:

```

8869 \newcommand*{\glsentrypluralaccess}[1]{%
8870   \csname glo@#1@pluralaccess\endcsname
8871 }

```

`\glsentryfirstpluralaccess` Get the value of the firstpluralaccess key for the entry with the given label:

```

8872 \newcommand*{\glsentryfirstpluralaccess}[1]{%
8873   \csname glo@#1@firstpluralaccess\endcsname
8874 }

```

`\glsentrysymbolaccess` Get the value of the symbolaccess key for the entry with the given label:

```

8875 \newcommand*{\glsentrysymbolaccess}[1]{%
8876   \csname glo@#1@symbolaccess\endcsname
8877 }

```

`\glsentrysymbolpluralaccess` Get the value of the symbolpluralaccess key for the entry with the given label:

```

8878 \newcommand*{\glsentrysymbolpluralaccess}[1]{%
8879   \csname glo@#1@symbolpluralaccess\endcsname
8880 }

```

`\glsentrydescaccess` Get the value of the descriptionaccess key for the entry with the given label:

```

8881 \newcommand*{\glsentrydescaccess}[1]{%
8882   \csname glo@#1@descaccess\endcsname
8883 }

```

`\glsentrydescpluralaccess` Get the value of the descriptionpluralaccess key for the entry with the given label:

```

8884 \newcommand*{\glsentrydescpluralaccess}[1]{%
8885   \csname glo@#1@descaccess\endcsname
8886 }

```

`\glsentryshortaccess` Get the value of the shortaccess key for the entry with the given label:

```

8887 \newcommand*{\glsentryshortaccess}[1]{%
8888   \csname glo@#1@shortaccess\endcsname
8889 }

```

`\glsentryshortpluralaccess` Get the value of the shortpluralaccess key for the entry with the given label:

```

8890 \newcommand*{\glsentryshortpluralaccess}[1]{%
8891   \csname glo@#1@shortpluralaccess\endcsname
8892 }

```

`\glsentrylongaccess` Get the value of the longaccess key for the entry with the given label:

```

8893 \newcommand*{\glsentrylongaccess}[1]{%
8894   \csname glo@#1@longaccess\endcsname
8895 }

```

`\glsentrylongpluralaccess` Get the value of the longpluralaccess key for the entry with the given label:

```

8896 \newcommand*{\glsentrylongpluralaccess}[1]{%
8897   \csname glo@#1@longpluralaccess\endcsname
8898 }

```

`\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.)

```

8899 \newcommand*{\glsaccsupp}[2]{%
8900   \BeginAccSupp{ActualText=#1}#2\EndAccSupp{}%
8901 }

```

`\xglsaccsupp` Fully expands replacement text before calling `\glsaccsupp`

```

8902 \newcommand*{\xglsaccsupp}[2]{%
8903   \protected@edef\@gls@replacementtext{#1}%
8904   \expandafter\glsaccsupp\expandafter{\@gls@replacementtext}{#2}%
8905 }

```

`\glsnameaccessdisplay` Displays the first argument with the accessibility text for the entry with the label given by the second argument (if set).

```

8906 \DeclareRobustCommand*{\glsnameaccessdisplay}[2]{%
8907   \protected@edef\@glo@access{\glsentryaccess{#2}}%
8908   \ifx\@glo@access\@gls@noaccess
8909     #1%
8910   \else
8911     \xglsaccsupp{\@glo@access}{#1}%
8912   \fi
8913 }

```

lstextaccessdisplay As above but for the textaccess replacement text.

```
8914 \DeclareRobustCommand*\glstextaccessdisplay}[2]{%
8915   \protected@edef\@glo@access{\glstentrytextaccess{#2}}%
8916   \ifx\@glo@access\@gls@noaccess
8917     #1%
8918   \else
8919     \xglsaccsupp{\@glo@access}{#1}%
8920   \fi
8921 }
```

pluralaccessdisplay As above but for the pluralaccess replacement text.

```
8922 \DeclareRobustCommand*\glspluralaccessdisplay}[2]{%
8923   \protected@edef\@glo@access{\glsentrypluralaccess{#2}}%
8924   \ifx\@glo@access\@gls@noaccess
8925     #1%
8926   \else
8927     \xglsaccsupp{\@glo@access}{#1}%
8928   \fi
8929 }
```

firstaccessdisplay As above but for the firstaccess replacement text.

```
8930 \DeclareRobustCommand*\glsfirstaccessdisplay}[2]{%
8931   \protected@edef\@glo@access{\glsentryfirstaccess{#2}}%
8932   \ifx\@glo@access\@gls@noaccess
8933     #1%
8934   \else
8935     \xglsaccsupp{\@glo@access}{#1}%
8936   \fi
8937 }
```

pluralfirstaccessdisplay As above but for the firstpluralaccess replacement text.

```
8938 \DeclareRobustCommand*\glsfirstpluralaccessdisplay}[2]{%
8939   \protected@edef\@glo@access{\glsentryfirstpluralaccess{#2}}%
8940   \ifx\@glo@access\@gls@noaccess
8941     #1%
8942   \else
8943     \xglsaccsupp{\@glo@access}{#1}%
8944   \fi
8945 }
```

symbolaccessdisplay As above but for the symbolaccess replacement text.

```
8946 \DeclareRobustCommand*\glssymbolaccessdisplay}[2]{%
8947   \protected@edef\@glo@access{\glsentrysymbolaccess{#2}}%
8948   \ifx\@glo@access\@gls@noaccess
8949     #1%
8950   \else
8951     \xglsaccsupp{\@glo@access}{#1}%
8952   \fi
8953 }
```

pluralaccessdisplay As above but for the symbolpluralaccess replacement text.

```
8954 \DeclareRobustCommand*\glssymbolpluralaccessdisplay}[2]{%
8955   \protected@edef\@glo@access{\glstrysymbolpluralaccess{#2}}%
8956   \ifx\@glo@access\@gls@noaccess
8957     #1%
8958   \else
8959     \xglsaccsupp{\@glo@access}{#1}%
8960   \fi
8961 }
```

descriptionaccessdisplay As above but for the descriptionaccess replacement text.

```
8962 \DeclareRobustCommand*\glsdescriptionaccessdisplay}[2]{%
8963   \protected@edef\@glo@access{\glstrydescaccess{#2}}%
8964   \ifx\@glo@access\@gls@noaccess
8965     #1%
8966   \else
8967     \xglsaccsupp{\@glo@access}{#1}%
8968   \fi
8969 }
```

pluralaccessdisplay As above but for the descriptionpluralaccess replacement text.

```
8970 \DeclareRobustCommand*\glsdescriptionpluralaccessdisplay}[2]{%
8971   \protected@edef\@glo@access{\glstrydescpluralaccess{#2}}%
8972   \ifx\@glo@access\@gls@noaccess
8973     #1%
8974   \else
8975     \xglsaccsupp{\@glo@access}{#1}%
8976   \fi
8977 }
```

shortaccessdisplay As above but for the shortaccess replacement text.

```
8978 \DeclareRobustCommand*\glsshortaccessdisplay}[2]{%
8979   \protected@edef\@glo@access{\glstryshortaccess{#2}}%
8980   \ifx\@glo@access\@gls@noaccess
8981     #1%
8982   \else
8983     \xglsaccsupp{\@glo@access}{#1}%
8984   \fi
8985 }
```

pluralaccessdisplay As above but for the shortpluralaccess replacement text.

```
8986 \DeclareRobustCommand*\glsshortpluralaccessdisplay}[2]{%
8987   \protected@edef\@glo@access{\glstryshortpluralaccess{#2}}%
8988   \ifx\@glo@access\@gls@noaccess
8989     #1%
8990   \else
8991     \xglsaccsupp{\@glo@access}{#1}%
8992   \fi
8993 }
```

`\glslongaccessdisplay` As above but for the `longaccess` replacement text.

```
8994 \DeclareRobustCommand*\glslongaccessdisplay}[2]{%
8995   \protected@edef\@glo@access{\glsentrylongaccess{#2}}%
8996   \ifx\@glo@access\@gls@noaccess
8997     #1%
8998   \else
8999     \xglsaccsupp{\@glo@access}{#1}%
9000   \fi
9001 }
```

`\glspluralaccessdisplay` As above but for the `longpluralaccess` replacement text.

```
9002 \DeclareRobustCommand*\glslongpluralaccessdisplay}[2]{%
9003   \protected@edef\@glo@access{\glsentrylongpluralaccess{#2}}%
9004   \ifx\@glo@access\@gls@noaccess
9005     #1%
9006   \else
9007     \xglsaccsupp{\@glo@access}{#1}%
9008   \fi
9009 }
```

`\glsaccessdisplay` Gets the replacement text corresponding to the named key given by the first argument and calls the appropriate command defined above.

```
9010 \DeclareRobustCommand*\glsaccessdisplay}[3]{%
9011   \@ifundefined{gls#1accessdisplay}%
9012   {%
9013     \PackageError{glossaries-accsupp}{No accessibility support
9014       for key ‘#1’}{}%
9015   }%
9016   {%
9017     \csname gls#1accessdisplay\endcsname{#2}{#3}%
9018   }%
9019 }
```

`\gls@default@entryfmt` Redefine the default entry format to use accessibility information

```
9020 \renewcommand*\@@gls@default@entryfmt}[2]{%
9021   \ifdefempty\gls@customtext
9022   {%
9023     \glsifplural
9024     {%
```

Plural form

```
9025     \glscapscase
9026     {%
```

Don't adjust case

```
9027     \ifglsused\glslabel
9028     {%
```

Subsequent use

```
9029     #2{\glspluralaccessdisplay
```

```

9030          {\glentryplural{\glslabel}}{\glslabel}}%
9031      {\glsdescriptionpluralaccessdisplay
9032          {\glentrydescplural{\glslabel}}{\glslabel}}%
9033      {\glssymbolpluralaccessdisplay
9034          {\glentrysymbolplural{\glslabel}}{\glslabel}}
9035      {\glsinsert}}%
9036  }%
9037  {%

```

First use

```

9038      #1{\glsfirstpluralaccessdisplay
9039          {\glentryfirstplural{\glslabel}}{\glslabel}}%
9040      {\glsdescriptionpluralaccessdisplay
9041          {\glentrydescplural{\glslabel}}{\glslabel}}%
9042      {\glssymbolpluralaccessdisplay
9043          {\glentrysymbolplural{\glslabel}}{\glslabel}}%
9044      {\glsinsert}}%
9045  }%
9046  }%
9047  {%

```

Make first letter upper case

```

9048      \ifglsused\glslabel
9049      {%

```

Subsequent use.

```

9050      #2{\glspluralaccessdisplay
9051          {\Glsentryplural{\glslabel}}{\glslabel}}%
9052      {\glsdescriptionpluralaccessdisplay
9053          {\glentrydescplural{\glslabel}}{\glslabel}}%
9054      {\glssymbolpluralaccessdisplay
9055          {\glentrysymbolplural{\glslabel}}{\glslabel}}%
9056      {\glsinsert}}%
9057  }%
9058  {%

```

First use

```

9059      #1{\glsfirstpluralaccessdisplay
9060          {\Glsentryfirstplural{\glslabel}}{\glslabel}}%
9061      {\glsdescriptionpluralaccessdisplay
9062          {\glentrydescplural{\glslabel}}{\glslabel}}%
9063      {\glssymbolpluralaccessdisplay
9064          {\glentrysymbolplural{\glslabel}}{\glslabel}}%
9065      {\glsinsert}}%
9066  }%
9067  }%
9068  {%

```

Make all upper case

```

9069      \ifglsused\glslabel
9070      {%

```

Subsequent use

```

9071      \MakeUppercase{%
9072          #2{\glspluralaccessdisplay
9073              {\glsentryplural{\glslabel}}{\glslabel}}%
9074              {\glsdescriptionpluralaccessdisplay
9075                  {\glsentrydescplural{\glslabel}}{\glslabel}}%
9076              {\glsymbolpluralaccessdisplay
9077                  {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
9078              {\glsinsert}}}%
9079      }%
9080      {%

```

First use

```

9081      \MakeUppercase{%
9082          #1{\glsfirstpluralaccessdisplay
9083              {\glsentryfirstplural{\glslabel}}{\glslabel}}%
9084              {\glsdescriptionpluralaccessdisplay
9085                  {\glsentrydescplural{\glslabel}}{\glslabel}}%
9086              {\glsymbolpluralaccessdisplay
9087                  {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
9088              {\glsinsert}}}%
9089      }%
9090      }%
9091      }%
9092      {%

```

Singular form

```

9093      \glscapscase
9094      {%

```

Don't adjust case

```

9095      \ifglsused\glslabel
9096      {%

```

Subsequent use

```

9097      #2{\glstextaccessdisplay
9098          {\glsentrytext{\glslabel}}{\glslabel}}%
9099          {\glsdescriptionaccessdisplay
9100              {\glsentrydesc{\glslabel}}{\glslabel}}%
9101              {\glsymbolaccessdisplay
9102                  {\glsentrysymbol{\glslabel}}{\glslabel}}%
9103              {\glsinsert}}%
9104      }%
9105      {%

```

First use

```

9106      #1{\glsfirstaccessdisplay
9107          {\glsentryfirst{\glslabel}}{\glslabel}}%
9108          {\glsdescriptionaccessdisplay
9109              {\glsentrydesc{\glslabel}}{\glslabel}}%
9110              {\glsymbolaccessdisplay

```

```

9111          {\glentrysymbol{\glslabel}}{\glslabel}}%
9112      {\glsinsert}%
9113  }%
9114  }%
9115  {%

```

Make first letter upper case

```

9116      \ifglsused\glslabel
9117      {%

```

Subsequent use

```

9118      #2{\glstextaccessdisplay
9119          {\Glsentrytext{\glslabel}}{\glslabel}}%
9120      {\glsdescriptionaccessdisplay
9121          {\glentrydesc{\glslabel}}{\glslabel}}%
9122      {\glssymbolaccessdisplay
9123          {\glentrysymbol{\glslabel}}{\glslabel}}%
9124      {\glsinsert}%
9125  }%
9126  {%

```

First use

```

9127      #1{\glsfirstaccessdisplay
9128          {\Glsentryfirst{\glslabel}}{\glslabel}}%
9129      {\glsdescriptionaccessdisplay
9130          {\glentrydesc{\glslabel}}{\glslabel}}%
9131      {\glssymbolaccessdisplay
9132          {\glentrysymbol{\glslabel}}{\glslabel}}%
9133      {\glsinsert}%
9134  }%
9135  }%
9136  {%

```

Make all upper case

```

9137      \ifglsused\glslabel
9138      {%

```

Subsequent use

```

9139      \MakeUppercase{%
9140      #2{\glstextaccessdisplay
9141          {\glentrytext{\glslabel}}{\glslabel}}%
9142      {\glsdescriptionaccessdisplay
9143          {\glentrydesc{\glslabel}}{\glslabel}}%
9144      {\glssymbolaccessdisplay
9145          {\glentrysymbol{\glslabel}}{\glslabel}}%
9146      {\glsinsert}}%
9147  }%
9148  {%

```

First use

```

9149      \MakeUppercase{%
9150      #1{\glsfirstaccessdisplay

```

```

9151         {\glsentryfirst{\glslabel}}{\glslabel}}%
9152     {\glsdescriptionaccessdisplay
9153         {\glsentrydesc{\glslabel}}{\glslabel}}%
9154     {\glssymbolaccessdisplay
9155         {\glsentrysymbol{\glslabel}}{\glslabel}}%
9156     {\glsinsert}}%
9157 }%
9158 }%
9159 }%
9160 }%
9161 {%

```

Custom text provided in \glsdisp

```

9162     \ifglsused{\glslabel}%
9163     {%

```

Subsequent use

```

9164         #2{\glscustomtext}%
9165         {\glsdescriptionaccessdisplay
9166             {\glsentrydesc{\glslabel}}{\glslabel}}%
9167         {\glssymbolaccessdisplay
9168             {\glsentrysymbol{\glslabel}}{\glslabel}}%
9169         {\glsinsert}}%
9170     }%
9171     {%

```

First use

```

9172         #1{\glscustomtext}%
9173         {\glsdescriptionaccessdisplay
9174             {\glsentrydesc{\glslabel}}{\glslabel}}%
9175         {\glssymbolaccessdisplay
9176             {\glsentrysymbol{\glslabel}}{\glslabel}}%
9177         {\glsinsert}}%
9178     }%
9179 }%
9180 }

```

\@acrshort

```

9181 \def\@acrshort#1#2[#3]{%
9182     \glsdoifexists{#2}%
9183     {%
9184         \edef\@glo@type{\glsentrytype{#2}}%
9185         \def\glslabel{#2}%
9186         \let\glsifplural\@secondoftwo
9187         \let\glscapscase\@firstofthree
9188         \let\glsinsert\@empty
9189         \def\glscustomtext{%
9190             \acronymfont{\glsshortaccessdisplay{\glsentryshort{#2}}{#2}}#3%
9191         }%

```

```

Call \@gls@link
9192   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
9193   }%
9194 }

```

\@Acrshort

```

9195 \def\@Acrshort#1#2[#3]{%
9196   \glsdoifexists{#2}%
9197   {%
9198     \edef\@glo@type{\glentrytype{#2}}%

9199     \def\glslabel{#2}%
9200     \let\glsifplural\@secondoftwo
9201     \let\glscapscase\@secondofthree
9202     \let\glsinsert\@empty
9203     \def\glscustomtext{%
9204       \acronymfont{\glsshortaccessdisplay{\Glsentryshort{#2}}{#2}}#3%
9205     }%

```

```

Call \@gls@link
9206   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
9207   }%
9208 }

```

\@ACRshort

```

9209 \def\@ACRshort#1#2[#3]{%
9210   \glsdoifexists{#2}%
9211   {%
9212     \edef\@glo@type{\glentrytype{#2}}%

9213     \def\glslabel{#2}%
9214     \let\glsifplural\@secondoftwo
9215     \let\glscapscase\@thirdofthree
9216     \let\glsinsert\@empty
9217     \def\glscustomtext{%
9218       \acronymfont{\glsshortaccessdisplay
9219         {\MakeUppercase{\glentryshort{#2}}}{#2}}#3%
9220     }%

```

```

Call \@gls@link
9221   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
9222   }%
9223 }

```

\@acrlong

```

9224 \def\@acrlong#1#2[#3]{%
9225   \glsdoifexists{#2}%
9226   {%
9227     \edef\@glo@type{\glentrytype{#2}}%

```

```

9228 \def\glslabel{#2}%
9229 \let\glsifplural\@secondoftwo
9230 \let\glscapscase\@firstofthree
9231 \let\glsinsert\@empty
9232 \def\glscustomtext{%
9233 \acronymfont{\glslongaccessdisplay{\glsentrylong{#2}}{#2}}#3%
9234 }%

Call \@gls@link
9235 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
9236 }%
9237 }

```

\@Acrlong

```

9238 \def\@Acrlong#1#2[#3]{%
9239 \glsdoifexists{#2}%
9240 {%
9241 \edef\@glo@type{\glsentrytype{#2}}%

9242 \def\glslabel{#2}%
9243 \let\glsifplural\@secondoftwo
9244 \let\glscapscase\@firstofthree
9245 \let\glsinsert\@empty
9246 \def\glscustomtext{%
9247 \acronymfont{\glslongaccessdisplay{\Glsentrylong{#2}}{#2}}#3%
9248 }%

Call \@gls@link
9249 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
9250 }%
9251 }

```

\@ACRlong

```

9252 \def\@ACRlong#1#2[#3]{%
9253 \glsdoifexists{#2}%
9254 {%
9255 \edef\@glo@type{\glsentrytype{#2}}%

9256 \def\glslabel{#2}%
9257 \let\glsifplural\@secondoftwo
9258 \let\glscapscase\@firstofthree
9259 \let\glsinsert\@empty
9260 \def\glscustomtext{%
9261 \acronymfont{\glslongaccessdisplay{%
9262 \MakeUppercase{\glsentrylong{#2}}}{#2}}#3%
9263 }%

Call \@gls@link
9264 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
9265 }%
9266 }

```

6.3 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.

```
9267 \renewcommand*{\glossentryname}[1]{%
9268   \glsdoifexists{#1}%
9269   {%
9270     \glsnamefont{\glsnameaccessdisplay{\glsentryname{#1}}{#1}}%
9271   }%
9272 }

9273 \renewcommand*{\glossentryname}[1]{%
9274   \glsdoifexists{#1}%
9275   {%
9276     \glsnamefont{\glsnameaccessdisplay{\Glsentryname{#1}}{#1}}%
9277   }%
9278 }

9279 \renewcommand*{\glossentrydesc}[1]{%
9280   \glsdoifexists{#1}%
9281   {%
9282     \glsdescriptionaccessdisplay{\glsentrydesc{#1}}{#1}%
9283   }%
9284 }

9285 \renewcommand*{\Glossentrydesc}[1]{%
9286   \glsdoifexists{#1}%
9287   {%
9288     \glsdescriptionaccessdisplay{\Glsentrydesc{#1}}{#1}%
9289   }%
9290 }

9291 \renewcommand*{\glossentrysymbol}[1]{%
9292   \glsdoifexists{#1}%
9293   {%
9294     \glssymbolaccessdisplay{\glsentrysymbol{#1}}{#1}%
9295   }%
9296 }

9297 \renewcommand*{\Glossentrysymbol}[1]{%
9298   \glsdoifexists{#1}%
9299   {%
9300     \glssymbolaccessdisplay{\Glsentrysymbol{#1}}{#1}%
9301   }%
9302 }
```

pglossaryentryfield

```

9303 \newcommand*{\accsuppglossaryentryfield}[5]{%
9304   \glossaryentryfield{#1}%
9305   {\glsnameaccessdisplay{#2}{#1}}%
9306   {\glsdescriptionaccessdisplay{#3}{#1}}%
9307   {\glsymbolaccessdisplay{#4}{#1}}{#5}%
9308 }

```

glossarysubentryfield

```

9309 \newcommand*{\accsuppglossarysubentryfield}[6]{%
9310   \glossarysubentryfield{#1}{#2}%
9311   {\glsnameaccessdisplay{#3}{#2}}%
9312   {\glsdescriptionaccessdisplay{#4}{#2}}%
9313   {\glsymbolaccessdisplay{#5}{#2}}{#6}%
9314 }

```

6.4 Acronyms

Use `\newacronymhook` to modify the key list to set the access text to the long version by default.

```

9315 \renewcommand*{\newacronymhook}{%
9316   \edef\@gls@keylist{shortaccess=\the\glslongtok,%
9317     \the\glskeylisttok}%
9318   \expandafter\glskeylisttok\expandafter{\@gls@keylist}%
9319 }

```

defaultNewAcronymDef Modify default style to use access text:

```

9320 \renewcommand*{\DefaultNewAcronymDef}{%
9321   \edef\@do@newglossaryentry{%
9322     \noexpand\newglossaryentry{\the\glslabeltok}%
9323     {%
9324       type=\acronymtype,%
9325       name={\the\glsshorttok},%
9326       description={\the\glslongtok},%
9327       descriptionaccess=\relax,%
9328       text={\the\glsshorttok},%
9329       access={\noexpand\@glo@textaccess},%
9330       sort={\the\glsshorttok},%
9331       short={\the\glsshorttok},%
9332       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
9333       shortaccess={\the\glslongtok},%
9334       long={\the\glslongtok},%
9335       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
9336       descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
9337       first={\noexpand\glslongaccessdisplay
9338         {\the\glslongtok}{\the\glslabeltok}\space
9339         {\noexpand\glsshortaccessdisplay
9340           {\the\glsshorttok}{\the\glslabeltok}}},%

```

```

9341 plural={\the\glsshorttok\acrpluralsuffix},%
9342 firstplural={\noexpand\glslongpluralaccessdisplay
9343   {\noexpand\@glo@longpl}{\the\glslabeltok}\space
9344   (\noexpand\glsshortpluralaccessdisplay
9345     {\noexpand\@glo@shortpl}{\the\glslabeltok})},%
9346 firstaccess=\relax,
9347 firstpluralaccess=\relax,
9348 textaccess={\noexpand\@glo@shortaccess},%
9349 \the\glskeylisttok
9350 }%
9351 }%
9352 \@do@newglossaryentry
9353 }

```

otnoteNewAcronymDef

```

9354 \renewcommand*{\DescriptionFootnoteNewAcronymDef}{%
9355   \edef\@do@newglossaryentry{%
9356     \noexpand\newglossaryentry{\the\glslabeltok}%
9357     {%
9358       type=\acronymtype,%
9359       name={\noexpand\acronymfont{\the\glsshorttok}},%
9360       sort={\the\glsshorttok},%
9361       text={\the\glsshorttok},%
9362       short={\the\glsshorttok},%
9363       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
9364       shortaccess={\the\glslongtok},%
9365       long={\the\glslongtok},%
9366       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
9367       access={\noexpand\@glo@textaccess},%
9368       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
9369       symbol={\the\glslongtok},%
9370       symbolplural={\the\glslongtok\noexpand\acrpluralsuffix},%
9371       firstpluralaccess=\relax,
9372       textaccess={\noexpand\@glo@shortaccess},%
9373       \the\glskeylisttok
9374     }%
9375   }%
9376   \@do@newglossaryentry
9377 }

```

iptionNewAcronymDef

```

9378 \renewcommand*{\DescriptionNewAcronymDef}{%
9379   \edef\@do@newglossaryentry{%
9380     \noexpand\newglossaryentry{\the\glslabeltok}%
9381     {%
9382       type=\acronymtype,%
9383       name={\noexpand
9384         \acrnameformat{\the\glsshorttok}{\the\glslongtok}},%
9385       access={\noexpand\@glo@textaccess},%

```

```

9386     sort={\the\glsshorttok},%
9387     short={\the\glsshorttok},%
9388     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
9389     shortaccess={\the\glslongtok},%
9390     long={\the\glslongtok},%
9391     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
9392     first={\the\glslongtok},%
9393     firstaccess=\relax,
9394     firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
9395     text={\the\glsshorttok},%
9396     textaccess={\the\glslongtok},%
9397     plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
9398     symbol={\noexpand\@glo@text},%
9399     symbolaccess={\noexpand\@glo@textaccess},%
9400     symbolplural={\noexpand\@glo@plural},%
9401     firstpluralaccess=\relax,
9402     textaccess={\noexpand\@glo@shortaccess},%
9403     \the\glskeylisttok}%
9404 }%
9405 \@do@newglossaryentry
9406 }

```

otnoteNewAcronymDef

```

9407 \renewcommand*{\FootnoteNewAcronymDef}{%
9408   \edef\@do@newglossaryentry{%
9409     \noexpand\newglossaryentry{\the\glslabeltok}%
9410     {%
9411       type=\acronymtype,%
9412       name={\noexpand\acronymfont{\the\glsshorttok}},%
9413       sort={\the\glsshorttok},%
9414       text={\the\glsshorttok},%
9415       textaccess={\the\glslongtok},%
9416       access={\noexpand\@glo@textaccess},%
9417       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
9418       short={\the\glsshorttok},%
9419       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
9420       long={\the\glslongtok},%
9421       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
9422       description={\the\glslongtok},%
9423       descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
9424       \the\glskeylisttok
9425     }%
9426   }%
9427   \@do@newglossaryentry
9428 }

```

\SmallNewAcronymDef

```

9429 \renewcommand*{\SmallNewAcronymDef}{%
9430   \edef\@do@newglossaryentry{%

```

```

9431 \noexpand\newglossaryentry{\the\glslabeltok}%
9432 {%
9433     type=\acronymtype,%
9434     name={\noexpand\acronymfont{\the\glsshorttok}},%
9435     access={\noexpand\@glo@symbolaccess},%
9436     sort={\the\glsshorttok},%
9437     short={\the\glsshorttok},%
9438     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
9439     shortaccess={\the\glslongtok},%
9440     long={\the\glslongtok},%
9441     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
9442     text={\noexpand\@glo@short},%
9443     textaccess={\noexpand\@glo@shortaccess},%
9444     plural={\noexpand\@glo@shortpl},%
9445     first={\the\glslongtok},%
9446     firstaccess=\relax,%
9447     firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
9448     description={\noexpand\@glo@first},%
9449     descriptionplural={\noexpand\@glo@firstplural},%
9450     symbol={\the\glsshorttok},%
9451     symbolaccess={\the\glslongtok},%
9452     symbolplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
9453     \the\glskeylisttok
9454 }%
9455 }%
9456 \@do@newglossaryentry
9457 }

```

The following are kept for compatibility with versions before 3.0:

```

\glsshortaccesskey
9458 \newcommand*{\glsshortaccesskey}{\glsshortkey access}%

hortpluralaccesskey
9459 \newcommand*{\glsshortpluralaccesskey}{\glsshortpluralkey access}%

\glslongaccesskey
9460 \newcommand*{\glslongaccesskey}{\glslongkey access}%

longpluralaccesskey
9461 \newcommand*{\glslongpluralaccesskey}{\glslongpluralkey access}%

```

6.5 Debugging Commands

```

\showglongnameaccess
9462 \newcommand*{\showglongnameaccess}[1]{%
9463     \expandafter\show\csname glo@#1@textaccess\endcsname
9464 }

```

```

\showglotextaccess
9465 \newcommand*{\showglotextaccess}[1]{%
9466   \expandafter\show\csname glo@#1@textaccess\endcsname
9467 }

showglopluralaccess
9468 \newcommand*{\showglopluralaccess}[1]{%
9469   \expandafter\show\csname glo@#1@pluralaccess\endcsname
9470 }

\showglofirstaccess
9471 \newcommand*{\showglofirstaccess}[1]{%
9472   \expandafter\show\csname glo@#1@firstaccess\endcsname
9473 }

lofirstpluralaccess
9474 \newcommand*{\showglofirstpluralaccess}[1]{%
9475   \expandafter\show\csname glo@#1@firstpluralaccess\endcsname
9476 }

showglosymbolaccess
9477 \newcommand*{\showglosymbolaccess}[1]{%
9478   \expandafter\show\csname glo@#1@symbolaccess\endcsname
9479 }

osymbolpluralaccess
9480 \newcommand*{\showglosymbolpluralaccess}[1]{%
9481   \expandafter\show\csname glo@#1@symbolpluralaccess\endcsname
9482 }

\showglodescaccess
9483 \newcommand*{\showglodescaccess}[1]{%
9484   \expandafter\show\csname glo@#1@descaccess\endcsname
9485 }

glodescpluralaccess
9486 \newcommand*{\showglodescpluralaccess}[1]{%
9487   \expandafter\show\csname glo@#1@descpluralaccess\endcsname
9488 }

\showgloshortaccess
9489 \newcommand*{\showgloshortaccess}[1]{%
9490   \expandafter\show\csname glo@#1@shortaccess\endcsname
9491 }

loshortpluralaccess
9492 \newcommand*{\showgloshortpluralaccess}[1]{%
9493   \expandafter\show\csname glo@#1@shortpluralaccess\endcsname
9494 }

```

\showglolongaccess

```
9495 \newcommand*{\showglolongaccess}[1]{%
9496   \expandafter\show\csname glo@#1@longaccess\endcsname
9497 }
```

glolongpluralaccess

```
9498 \newcommand*{\showglolongpluralaccess}[1]{%
9499   \expandafter\show\csname glo@#1@longpluralaccess\endcsname
9500 }
```

7 Multi-Lingual Support

Many thanks to everyone who contributed to the translations both via email and on comp.text.tex.

7.1 Babel Captions

Define captions if multi-lingual support is required, but the package is not loaded.

```
9501 \NeedsTeXFormat{LaTeX2e}
9502 \ProvidesPackage{glossaries-babel}[2013/11/14 v4.0 (NLCT)]
```

English:

```
9503 \@ifundefined{captionsenglish}{}{%
9504   \addto\captionsenglish{%
9505     \renewcommand*{\glossaryname}{Glossary}%
9506     \renewcommand*{\acronymname}{Acronyms}%
9507     \renewcommand*{\entryname}{Notation}%
9508     \renewcommand*{\descriptionname}{Description}%
9509     \renewcommand*{\symbolname}{Symbol}%
9510     \renewcommand*{\pagelistname}{Page List}%
9511     \renewcommand*{\glssymbolsgroupname}{Symbols}%
9512     \renewcommand*{\glsnumbersgroupname}{Numbers}%
9513 }%
9514 }
9515 \@ifundefined{captionsamerican}{}{%
9516   \addto\captionsamerican{%
9517     \renewcommand*{\glossaryname}{Glossary}%
9518     \renewcommand*{\acronymname}{Acronyms}%
9519     \renewcommand*{\entryname}{Notation}%
9520     \renewcommand*{\descriptionname}{Description}%
9521     \renewcommand*{\symbolname}{Symbol}%
9522     \renewcommand*{\pagelistname}{Page List}%
9523     \renewcommand*{\glssymbolsgroupname}{Symbols}%
9524     \renewcommand*{\glsnumbersgroupname}{Numbers}%
9525 }%
9526 }
9527 \@ifundefined{captionsaustralian}{}{%
```

```

9528 \addto\captionsaustrian{%
9529 \renewcommand*{\glossaryname}{Glossary}%
9530 \renewcommand*{\acronymname}{Acronyms}%
9531 \renewcommand*{\entryname}{Notation}%
9532 \renewcommand*{\descriptionname}{Description}%
9533 \renewcommand*{\symbolname}{Symbol}%
9534 \renewcommand*{\pagelistname}{Page List}%
9535 \renewcommand*{\glssymbolsgroupname}{Symbols}%
9536 \renewcommand*{\glsnumbersgroupname}{Numbers}%
9537 }%
9538 }
9539 \@ifundefined{captionsbritish}{-}{%
9540 \addto\captionsbritish{%
9541 \renewcommand*{\glossaryname}{Glossary}%
9542 \renewcommand*{\acronymname}{Acronyms}%
9543 \renewcommand*{\entryname}{Notation}%
9544 \renewcommand*{\descriptionname}{Description}%
9545 \renewcommand*{\symbolname}{Symbol}%
9546 \renewcommand*{\pagelistname}{Page List}%
9547 \renewcommand*{\glssymbolsgroupname}{Symbols}%
9548 \renewcommand*{\glsnumbersgroupname}{Numbers}%
9549 }%
9550 \@ifundefined{captionscanadian}{-}{%
9551 \addto\captionscanadian{%
9552 \renewcommand*{\glossaryname}{Glossary}%
9553 \renewcommand*{\acronymname}{Acronyms}%
9554 \renewcommand*{\entryname}{Notation}%
9555 \renewcommand*{\descriptionname}{Description}%
9556 \renewcommand*{\symbolname}{Symbol}%
9557 \renewcommand*{\pagelistname}{Page List}%
9558 \renewcommand*{\glssymbolsgroupname}{Symbols}%
9559 \renewcommand*{\glsnumbersgroupname}{Numbers}%
9560 }%
9561 }
9562 \@ifundefined{captionsnewzealand}{-}{%
9563 \addto\captionnewzealand{%
9564 \renewcommand*{\glossaryname}{Glossary}%
9565 \renewcommand*{\acronymname}{Acronyms}%
9566 \renewcommand*{\entryname}{Notation}%
9567 \renewcommand*{\descriptionname}{Description}%
9568 \renewcommand*{\symbolname}{Symbol}%
9569 \renewcommand*{\pagelistname}{Page List}%
9570 \renewcommand*{\glssymbolsgroupname}{Symbols}%
9571 \renewcommand*{\glsnumbersgroupname}{Numbers}%
9572 }%
9573 }
9574 \@ifundefined{captionsUKenglish}{-}{%
9575 \addto\captionUKenglish{%
9576 \renewcommand*{\glossaryname}{Glossary}%

```

```

9577 \renewcommand*{\acronymname}{Acronyms}%
9578 \renewcommand*{\entryname}{Notation}%
9579 \renewcommand*{\descriptionname}{Description}%
9580 \renewcommand*{\symbolname}{Symbol}%
9581 \renewcommand*{\pagelistname}{Page List}%
9582 \renewcommand*{\glssymbolsgroupname}{Symbols}%
9583 \renewcommand*{\glsnumbersgroupname}{Numbers}%
9584 }%
9585 }
9586 \@ifundefined{captionsUSenglish}{-}{%
9587 \addto\captionsUSenglish{%
9588 \renewcommand*{\glossaryname}{Glossary}%
9589 \renewcommand*{\acronymname}{Acronyms}%
9590 \renewcommand*{\entryname}{Notation}%
9591 \renewcommand*{\descriptionname}{Description}%
9592 \renewcommand*{\symbolname}{Symbol}%
9593 \renewcommand*{\pagelistname}{Page List}%
9594 \renewcommand*{\glssymbolsgroupname}{Symbols}%
9595 \renewcommand*{\glsnumbersgroupname}{Numbers}%
9596 }%
9597 }

```

German (quite a few variations were suggested for German; I settled on the following):

```

9598 \@ifundefined{captionsgerman}{-}{%
9599 \addto\captionsgerman{%
9600 \renewcommand*{\glossaryname}{Glossar}%
9601 \renewcommand*{\acronymname}{Akronyme}%
9602 \renewcommand*{\entryname}{Bezeichnung}%
9603 \renewcommand*{\descriptionname}{Beschreibung}%
9604 \renewcommand*{\symbolname}{Symbol}%
9605 \renewcommand*{\pagelistname}{Seiten}%
9606 \renewcommand*{\glssymbolsgroupname}{Symbole}%
9607 \renewcommand*{\glsnumbersgroupname}{Zahlen}}
9608 }

```

ngerman is identical to German:

```

9609 \@ifundefined{captionsgerman}{-}{%
9610 \addto\captionsgerman{%
9611 \renewcommand*{\glossaryname}{Glossar}%
9612 \renewcommand*{\acronymname}{Akronyme}%
9613 \renewcommand*{\entryname}{Bezeichnung}%
9614 \renewcommand*{\descriptionname}{Beschreibung}%
9615 \renewcommand*{\symbolname}{Symbol}%
9616 \renewcommand*{\pagelistname}{Seiten}%
9617 \renewcommand*{\glssymbolsgroupname}{Symbole}%
9618 \renewcommand*{\glsnumbersgroupname}{Zahlen}}
9619 }

```

Italian:

```

9620 \@ifundefined{captionsitalian}{-}{%

```

```

9621 \addto\captionssitalian{%
9622   \renewcommand*{\glossaryname}{Glossario}%
9623   \renewcommand*{\acronymname}{Acronimi}%
9624   \renewcommand*{\entryname}{Nomenclatura}%
9625   \renewcommand*{\descriptionname}{Descrizione}%
9626   \renewcommand*{\symbolname}{Simbolo}%
9627   \renewcommand*{\pagelistname}{Elenco delle pagine}%
9628   \renewcommand*{\glssymbolsgroupname}{Simboli}%
9629   \renewcommand*{\glsnumbersgroupname}{Numeri}}
9630 }

```

Dutch:

```

9631 \@ifundefined{captionssdutch}{-}{%
9632   \addto\captionssdutch{%
9633     \renewcommand*{\glossaryname}{Woordenlijst}%
9634     \renewcommand*{\acronymname}{Acroniemen}%
9635     \renewcommand*{\entryname}{Benaming}%
9636     \renewcommand*{\descriptionname}{Beschrijving}%
9637     \renewcommand*{\symbolname}{Symbool}%
9638     \renewcommand*{\pagelistname}{Pagina's}%
9639     \renewcommand*{\glssymbolsgroupname}{Symbolen}%
9640     \renewcommand*{\glsnumbersgroupname}{Cijfers}}
9641 }

```

Spanish:

```

9642 \@ifundefined{captionssspanish}{-}{%
9643   \addto\captionssspanish{%
9644     \renewcommand*{\glossaryname}{Glosario}%
9645     \renewcommand*{\acronymname}{Siglas}%
9646     \renewcommand*{\entryname}{Entrada}%
9647     \renewcommand*{\descriptionname}{Descripci\'on}%
9648     \renewcommand*{\symbolname}{S\'mbolo}%
9649     \renewcommand*{\pagelistname}{Lista de p\'aginas}%
9650     \renewcommand*{\glssymbolsgroupname}{S\'mbolos}%
9651     \renewcommand*{\glsnumbersgroupname}{N\'umeros}}
9652 }

```

French:

```

9653 \@ifundefined{captionssfrench}{-}{%
9654   \addto\captionssfrench{%
9655     \renewcommand*{\glossaryname}{Glossaire}%
9656     \renewcommand*{\acronymname}{Acronymes}%
9657     \renewcommand*{\entryname}{Termes}%
9658     \renewcommand*{\descriptionname}{Description}%
9659     \renewcommand*{\symbolname}{Symbole}%
9660     \renewcommand*{\pagelistname}{Pages}%
9661     \renewcommand*{\glssymbolsgroupname}{Symboles}%
9662     \renewcommand*{\glsnumbersgroupname}{Nombres}}
9663 }
9664 \@ifundefined{captionssfrenchb}{-}{%
9665   \addto\captionssfrenchb{%

```

```

9666 \renewcommand*{\glossaryname}{Glossaire}%
9667 \renewcommand*{\acronymname}{Acronymes}%
9668 \renewcommand*{\entryname}{Terme}%
9669 \renewcommand*{\descriptionname}{Description}%
9670 \renewcommand*{\symbolname}{Symbole}%
9671 \renewcommand*{\pagelistname}{Pages}%
9672 \renewcommand*{\glssymbolsgroupname}{Symboles}%
9673 \renewcommand*{\glslnumbersgroupname}{Nombres}}
9674 }
9675 \@ifundefined{captionsfrançais}{%
9676 \addto\captionsfrançais{
9677 \renewcommand*{\glossaryname}{Glossaire}%
9678 \renewcommand*{\acronymname}{Acronymes}%
9679 \renewcommand*{\entryname}{Terme}%
9680 \renewcommand*{\descriptionname}{Description}%
9681 \renewcommand*{\symbolname}{Symbole}%
9682 \renewcommand*{\pagelistname}{Pages}%
9683 \renewcommand*{\glssymbolsgroupname}{Symboles}%
9684 \renewcommand*{\glslnumbersgroupname}{Nombres}}
9685 }

```

Danish:

```

9686 \@ifundefined{captionsdanish}{%
9687 \addto\captionsdanish{
9688 \renewcommand*{\glossaryname}{Ordliste}%
9689 \renewcommand*{\acronymname}{Akronymer}%
9690 \renewcommand*{\entryname}{Symbolforklaring}%
9691 \renewcommand*{\descriptionname}{Beskrivelse}%
9692 \renewcommand*{\symbolname}{Symbol}%
9693 \renewcommand*{\pagelistname}{Side}%
9694 \renewcommand*{\glssymbolsgroupname}{Symboler}%
9695 \renewcommand*{\glslnumbersgroupname}{Tal}}
9696 }

```

Irish:

```

9697 \@ifundefined{captionsirish}{%
9698 \addto\captionsirish{
9699 \renewcommand*{\glossaryname}{Gluais}%
9700 \renewcommand*{\acronymname}{Acrainmneacha}%

```

wasn't sure whether to go for Nóta (Note), Ciall ('Meaning', 'sense') or Brí ('Meaning'). In the end I chose Ciall.

```

9701 \renewcommand*{\entryname}{Ciall}%
9702 \renewcommand*{\descriptionname}{Tuairisc}%

```

Again, not sure whether to use Comhartha/Comharthaí or Siombail/Siombaile, so have chosen the former.

```

9703 \renewcommand*{\symbolname}{Comhartha}%
9704 \renewcommand*{\glssymbolsgroupname}{Comhartha\'}{\i}}%
9705 \renewcommand*{\pagelistname}{Leathanaigh}%
9706 \renewcommand*{\glslnumbersgroupname}{Uimhreacha}}

```

9707 }

Hungarian:

```
9708 \@ifundefined{captionsmagyar}{%}{%
9709   \addto\captionsmagyar{%
9710     \renewcommand*{\glossaryname}{Sz\`ojegyz\`ek}%
9711     \renewcommand*{\acronymname}{Bet\H uszavak}%
9712     \renewcommand*{\entryname}{Kifejez\`es}%
9713     \renewcommand*{\descriptionname}{Magyar\`azat}%
9714     \renewcommand*{\symbolname}{Jel\"ol\`es}%
9715     \renewcommand*{\pagelistname}{Oldalsz\`am}%
9716     \renewcommand*{\glssymbolsgroupname}{Jelek}%
9717     \renewcommand*{\glsnumbersgroupname}{Sz\`amjegyek}%
9718   }
9719 }
9720 \@ifundefined{captionshungarian}{%}{%
9721   \addto\captionshungarian{%
9722     \renewcommand*{\glossaryname}{Sz\`ojegyz\`ek}%
9723     \renewcommand*{\acronymname}{Bet\H uszavak}%
9724     \renewcommand*{\entryname}{Kifejez\`es}%
9725     \renewcommand*{\descriptionname}{Magyar\`azat}%
9726     \renewcommand*{\symbolname}{Jel\"ol\`es}%
9727     \renewcommand*{\pagelistname}{Oldalsz\`am}%
9728     \renewcommand*{\glssymbolsgroupname}{Jelek}%
9729     \renewcommand*{\glsnumbersgroupname}{Sz\`amjegyek}%
9730   }
9731 }
```

Polish

```
9732 \@ifundefined{captionspolish}{%}{%
9733   \addto\captionspolish{%
9734     \renewcommand*{\glossaryname}{S{\l}ownik termin\`ow}%
9735     \renewcommand*{\acronymname}{Skr\`ot}%
9736     \renewcommand*{\entryname}{Termin}%
9737     \renewcommand*{\descriptionname}{Opis}%
9738     \renewcommand*{\symbolname}{Symbol}%
9739     \renewcommand*{\pagelistname}{Strony}%
9740     \renewcommand*{\glssymbolsgroupname}{Symbole}%
9741     \renewcommand*{\glsnumbersgroupname}{Liczby}}
9742 }
```

Brazilian

```
9743 \@ifundefined{captionsbrazil}{%}{%
9744   \addto\captionsbrazil{%
9745     \renewcommand*{\glossaryname}{Gloss\`ario}%
9746     \renewcommand*{\acronymname}{Siglas}%
9747     \renewcommand*{\entryname}{Nota\c c\~ao}%
9748     \renewcommand*{\descriptionname}{Descri\c c\~ao}%
9749     \renewcommand*{\symbolname}{S\`imbolo}%
9750     \renewcommand*{\pagelistname}{Lista de P\`aginas}%
9751     \renewcommand*{\glssymbolsgroupname}{S\`imbolos}%

```

```

9752 \renewcommand*{\glsnumbersgroupname}{N\,umeros}%
9753 }%
9754 }

```

7.2 Polyglossia Captions

```

9755 \NeedsTeXFormat{LaTeX2e}
9756 \ProvidesPackage{glossaries-polyglossia}[2013/11/14 v4.0 (NLCT)]

```

English:

```

9757 \@ifundefined{captionsenglish}{}{%
9758 \expandafter\toks@\expandafter{\captionsenglish
9759 \renewcommand*{\glossaryname}{\textenglish{Glossary}}%
9760 \renewcommand*{\acronymname}{\textenglish{Acronyms}}%
9761 \renewcommand*{\entryname}{\textenglish{Notation}}%
9762 \renewcommand*{\descriptionname}{\textenglish{Description}}%
9763 \renewcommand*{\symbolname}{\textenglish{Symbol}}%
9764 \renewcommand*{\pagelistname}{\textenglish{Page List}}%
9765 \renewcommand*{\glssymbolsgroupname}{\textenglish{Symbols}}%
9766 \renewcommand*{\glsnumbersgroupname}{\textenglish{Numbers}}%
9767 }%
9768 \edef\captionsenglish{\the\toks@}%
9769 }

```

German:

```

9770 \@ifundefined{captionsgerman}{}{%
9771 \expandafter\toks@\expandafter{\captionsgerman
9772 \renewcommand*{\glossaryname}{\textgerman{Glossar}}%
9773 \renewcommand*{\acronymname}{\textgerman{Akronyme}}%
9774 \renewcommand*{\entryname}{\textgerman{Bezeichnung}}%
9775 \renewcommand*{\descriptionname}{\textgerman{Beschreibung}}%
9776 \renewcommand*{\symbolname}{\textgerman{Symbol}}%
9777 \renewcommand*{\pagelistname}{\textgerman{Seiten}}%
9778 \renewcommand*{\glssymbolsgroupname}{\textgerman{Symbole}}%
9779 \renewcommand*{\glsnumbersgroupname}{\textgerman{Zahlen}}%
9780 }%
9781 \edef\captionsgerman{\the\toks@}%
9782 }

```

Italian:

```

9783 \@ifundefined{captionsitalian}{}{%
9784 \expandafter\toks@\expandafter{\captionsitalian
9785 \renewcommand*{\glossaryname}{\textitalian{Glossario}}%
9786 \renewcommand*{\acronymname}{\textitalian{Acronimi}}%
9787 \renewcommand*{\entryname}{\textitalian{Nomenclatura}}%
9788 \renewcommand*{\descriptionname}{\textitalian{Descrizione}}%
9789 \renewcommand*{\symbolname}{\textitalian{Simbolo}}%
9790 \renewcommand*{\pagelistname}{\textitalian{Elenco delle pagine}}%
9791 \renewcommand*{\glssymbolsgroupname}{\textitalian{Simboli}}%
9792 \renewcommand*{\glsnumbersgroupname}{\textitalian{Numeri}}%
9793 }%

```

```

9794 \edef\captionsitalian{\the\toks@}%
9795 }

```

Dutch:

```

9796 \@ifundefined{captionsdutch}{}{%
9797 \expandafter\toks@\expandafter{\captionsdutch
9798 \renewcommand*{\glossaryname}{\textdutch{Woordenlijst}}%
9799 \renewcommand*{\acronymname}{\textdutch{Acroniemen}}%
9800 \renewcommand*{\entryname}{\textdutch{Benaming}}%
9801 \renewcommand*{\descriptionname}{\textdutch{Beschrijving}}%
9802 \renewcommand*{\symbolname}{\textdutch{Symbool}}%
9803 \renewcommand*{\pagelistname}{\textdutch{Pagina's}}%
9804 \renewcommand*{\glssymbolsgroupname}{\textdutch{Symbolen}}%
9805 \renewcommand*{\glsnumbersgroupname}{\textdutch{Cijfers}}%
9806 }%
9807 \edef\captionsdutch{\the\toks@}%
9808 }

```

Spanish:

```

9809 \@ifundefined{captionsspanish}{}{%
9810 \expandafter\toks@\expandafter{\captionsspanish
9811 \renewcommand*{\glossaryname}{\textspanish{Glosario}}%
9812 \renewcommand*{\acronymname}{\textspanish{Siglas}}%
9813 \renewcommand*{\entryname}{\textspanish{Entrada}}%
9814 \renewcommand*{\descriptionname}{\textspanish{Descripci'on}}%
9815 \renewcommand*{\symbolname}{\textspanish{S'\i mbolo}}%
9816 \renewcommand*{\pagelistname}{\textspanish{Lista de p'aginas}}%
9817 \renewcommand*{\glssymbolsgroupname}{\textspanish{S'\i mbolos}}%
9818 \renewcommand*{\glsnumbersgroupname}{\textspanish{N'umeros}}%
9819 }%
9820 \edef\captionsspanish{\the\toks@}%
9821 }

```

French:

```

9822 \@ifundefined{captionsfrench}{}{%
9823 \expandafter\toks@\expandafter{\captionsfrench
9824 \renewcommand*{\glossaryname}{\textfrench{Glossaire}}%
9825 \renewcommand*{\acronymname}{\textfrench{Acronymes}}%
9826 \renewcommand*{\entryname}{\textfrench{Terme}}%
9827 \renewcommand*{\descriptionname}{\textfrench{Description}}%
9828 \renewcommand*{\symbolname}{\textfrench{Symbole}}%
9829 \renewcommand*{\pagelistname}{\textfrench{Pages}}%
9830 \renewcommand*{\glssymbolsgroupname}{\textfrench{Symboles}}%
9831 \renewcommand*{\glsnumbersgroupname}{\textfrench{Nombres}}%
9832 }%
9833 \edef\captionsfrench{\the\toks@}%
9834 }

```

Danish:

```

9835 \@ifundefined{captionsdanish}{}{%
9836 \expandafter\toks@\expandafter{\captionsdanish
9837 \renewcommand*{\glossaryname}{\textdanish{Ordliste}}%

```

```

9838 \renewcommand*{\acronymname}{\textdanish{Akronymer}}}%
9839 \renewcommand*{\entryname}{\textdanish{Symbolforklaring}}}%
9840 \renewcommand*{\descriptionname}{\textdanish{Beskrivelse}}}%
9841 \renewcommand*{\symbolname}{\textdanish{Symbol}}}%
9842 \renewcommand*{\pagelistname}{\textdanish{Side}}}%
9843 \renewcommand*{\glssymbolsgroupname}{\textdanish{Symboler}}}%
9844 \renewcommand*{\glsnumbersgroupname}{\textdanish{Tal}}}%
9845 }%
9846 \edef\captionsdanish{\the\toks@}%
9847 }

Irish:
9848 \@ifundefined{captionsirish}{}{%
9849 \expandafter\toks@\expandafter{\captionsirish
9850 \renewcommand*{\glossaryname}{\textirish{Gluais}}}%
9851 \renewcommand*{\acronymname}{\textirish{Acrainmneacha}}}%
9852 \renewcommand*{\entryname}{\textirish{Ciall}}}%
9853 \renewcommand*{\descriptionname}{\textirish{Tuirisc}}}%
9854 \renewcommand*{\symbolname}{\textirish{Comhartha}}}%
9855 \renewcommand*{\glssymbolsgroupname}{\textirish{Comhartha\,'{i}}}%
9856 \renewcommand*{\pagelistname}{\textirish{Leathanaigh}}}%
9857 \renewcommand*{\glsnumbersgroupname}{\textirish{Uimhreacha}}}%
9858 }%
9859 \edef\captionsirish{\the\toks@}%
9860 }

Hungarian:
9861 \@ifundefined{captionsmagyar}{}{%
9862 \expandafter\toks@\expandafter{\captionsmagyar
9863 \renewcommand*{\glossaryname}{\textmagyar{Sz\,'ojegyz\,'ek}}}%
9864 \renewcommand*{\acronymname}{\textmagyar{Bet\H uszavak}}}%
9865 \renewcommand*{\entryname}{\textmagyar{Kifejez\,'es}}}%
9866 \renewcommand*{\descriptionname}{\textmagyar{Magyar\,'azat}}}%
9867 \renewcommand*{\symbolname}{\textmagyar{Jel\,"ol\,'es}}}%
9868 \renewcommand*{\pagelistname}{\textmagyar{Oldalsz\,'am}}}%
9869 \renewcommand*{\glssymbolsgroupname}{\textmagyar{Jelek}}}%
9870 \renewcommand*{\glsnumbersgroupname}{\textmagyar{Sz\,'amjegyek}}}%
9871 }%
9872 \edef\captionsmagyar{\the\toks@}%
9873 }

Polish
9874 \@ifundefined{captionspolish}{}{%
9875 \expandafter\toks@\expandafter{\captionspolish
9876 \renewcommand*{\glossaryname}{\textpolish{S{\l}ownik termin\,'ow}}}%
9877 \renewcommand*{\acronymname}{\textpolish{Skr\,'ot}}}%
9878 \renewcommand*{\entryname}{\textpolish{Termin}}}%
9879 \renewcommand*{\descriptionname}{\textpolish{Opis}}}%
9880 \renewcommand*{\symbolname}{\textpolish{Symbol}}}%
9881 \renewcommand*{\pagelistname}{\textpolish{Strony}}}%
9882 \renewcommand*{\glssymbolsgroupname}{\textpolish{Symbole}}}%

```

```

9883 \renewcommand*{\glsnumbersgroupname}{\textpolish{Liczby}}%
9884 }%
9885 \edef\captionspolish{\the\toks@}%
9886 }

```

Portugues

```

9887 \@ifundefined{captionsportuges}{}{%
9888 \expandafter\toks@\expandafter{\captionsportuges
9889 \renewcommand*{\glossaryname}{\textportuges{Gloss\'ario}}%
9890 \renewcommand*{\acronymname}{\textportuges{Siglas}}%
9891 \renewcommand*{\entryname}{\textportuges{Nota\c c\~ao}}%
9892 \renewcommand*{\descriptionname}{\textportuges{Descri\c c\~ao}}%
9893 \renewcommand*{\symbolname}{\textportuges{S\'imbolo}}%
9894 \renewcommand*{\pagelistname}{\textportuges{Lista de P\'aginas}}%
9895 \renewcommand*{\glssymbolsgroupname}{\textportuges{S\'imbolos}}%
9896 \renewcommand*{\glsnumbersgroupname}{\textportuges{N\'umeros}}%
9897 }%
9898 \edef\captionsportuges{\the\toks@}%
9899 }

```

7.3 Brazilian Dictionary

This is a dictionary file provided by Thiago de Melo for use with the package.

```

9900 \ProvidesDictionary{glossaries-dictionary}{Brazilian}

```

Provide Brazilian translations:

```

9901 \providetranslation{Glossary}{Gloss\'ario}
9902 \providetranslation{Acronyms}{Siglas}
9903 \providetranslation{Notation (glossaries)}{Nota\c c\~ao}
9904 \providetranslation{Description (glossaries)}{Descri\c c\~ao}
9905 \providetranslation{Symbol (glossaries)}{S\'imbolo}
9906 \providetranslation{Page List (glossaries)}{Lista de P\'aginas}
9907 \providetranslation{Symbols (glossaries)}{S\'imbolos}
9908 \providetranslation{Numbers (glossaries)}{N\'umeros}

```

7.4 Danish Dictionary

This is a dictionary file provided for use with the package.

```

9909 \ProvidesDictionary{glossaries-dictionary}{Danish}

```

Provide Danish translations:

```

9910 \providetranslation{Glossary}{Ordliste}
9911 \providetranslation{Acronyms}{Akronymer}
9912 \providetranslation{Notation (glossaries)}{Symbolforklaring}
9913 \providetranslation{Description (glossaries)}{Beskrivelse}
9914 \providetranslation{Symbol (glossaries)}{Symbol}
9915 \providetranslation{Page List (glossaries)}{Side}
9916 \providetranslation{Symbols (glossaries)}{Symboler}
9917 \providetranslation{Numbers (glossaries)}{Tal}

```

7.5 Dutch Dictionary

This is a dictionary file provided for use with the package.

```
9918 \ProvidesDictionary{glossaries-dictionary}{Dutch}
```

Provide Dutch translations:

```
9919 \providetranslation{Glossary}{Woordenlijst}
9920 \providetranslation{Acronyms}{Acroniemen}
9921 \providetranslation{Notation (glossaries)}{Benaming}
9922 \providetranslation{Description (glossaries)}{Beschrijving}
9923 \providetranslation{Symbol (glossaries)}{Symbool}
9924 \providetranslation{Page List (glossaries)}{Pagina's}
9925 \providetranslation{Symbols (glossaries)}{Symbolen}
9926 \providetranslation{Numbers (glossaries)}{Cijfers}
```

7.6 English Dictionary

This is a dictionary file provided for use with the package.

```
9927 \ProvidesDictionary{glossaries-dictionary}{English}
```

Provide English translations:

```
9928 \providetranslation{Glossary}{Glossary}
9929 \providetranslation{Acronyms}{Acronyms}
9930 \providetranslation{Notation (glossaries)}{Notation}
9931 \providetranslation{Description (glossaries)}{Description}
9932 \providetranslation{Symbol (glossaries)}{Symbol}
9933 \providetranslation{Page List (glossaries)}{Page List}
9934 \providetranslation{Symbols (glossaries)}{Symbols}
9935 \providetranslation{Numbers (glossaries)}{Numbers}
```

7.7 French Dictionary

This is a dictionary file provided for use with the package.

```
9936 \ProvidesDictionary{glossaries-dictionary}{French}
```

Provide French translations:

```
9937 \providetranslation{Glossary}{Glossaire}
9938 \providetranslation{Acronyms}{Acronymes}
9939 \providetranslation{Notation (glossaries)}{Terme}
9940 \providetranslation{Description (glossaries)}{Description}
9941 \providetranslation{Symbol (glossaries)}{Symbole}
9942 \providetranslation{Page List (glossaries)}{Pages}
9943 \providetranslation{Symbols (glossaries)}{Symboles}
9944 \providetranslation{Numbers (glossaries)}{Nombres}
```

7.8 German Dictionary

This is a dictionary file provided for use with the package.

```
9945 \ProvidesDictionary{glossaries-dictionary}{German}
```

Provide German translations (quite a few variations were suggested for German; I settled on the following):

```
9946 \providetranslation{Glossary}{Glossar}
9947 \providetranslation{Acronyms}{Akronyme}
9948 \providetranslation{Notation (glossaries)}{Bezeichnung}
9949 \providetranslation{Description (glossaries)}{Beschreibung}
9950 \providetranslation{Symbol (glossaries)}{Symbol}
9951 \providetranslation{Page List (glossaries)}{Seiten}
9952 \providetranslation{Symbols (glossaries)}{Symbole}
9953 \providetranslation{Numbers (glossaries)}{Zahlen}
```

7.9 Irish Dictionary

This is a dictionary file provided for use with the package.

```
9954 \ProvidesDictionary{glossaries-dictionary}{Irish}
```

Provide Irish translations:

```
9955 \providetranslation{Glossary}{Gluais}
9956 \providetranslation{Acronyms}{Acrainmneacha}
9957 \providetranslation{Notation (glossaries)}{Ciall}
9958 \providetranslation{Description (glossaries)}{Tuairisc}
9959 \providetranslation{Symbol (glossaries)}{Comhartha}
9960 \providetranslation{Page List (glossaries)}{Leathanaigh}
9961 \providetranslation{Symbols (glossaries)}{Comhartha'\{i}}
9962 \providetranslation{Numbers (glossaries)}{Uimhreacha}
```

7.10 Italian Dictionary

This is a dictionary file provided for use with the package.

```
9963 \ProvidesDictionary{glossaries-dictionary}{Italian}
```

Provide Italian translations:

```
9964 \providetranslation{Glossary}{Glossario}
9965 \providetranslation{Acronyms}{Acronimi}
9966 \providetranslation{Notation (glossaries)}{Nomenclatura}
9967 \providetranslation{Description (glossaries)}{Descrizione}
9968 \providetranslation{Symbol (glossaries)}{Simbolo}
9969 \providetranslation{Page List (glossaries)}{Elenco delle pagine}
9970 \providetranslation{Symbols (glossaries)}{Simboli}
9971 \providetranslation{Numbers (glossaries)}{Numeri}
```

7.11 Magyar Dictionary

This is a dictionary file provided for use with the package.

```
9972 \ProvidesDictionary{glossaries-dictionary}{Magyar}
```

Provide translations:

```
9973 \providetranslation{Glossary}{Sz\'ojegyz\'ek}
9974 \providetranslation{Acronyms}{Bet\H uszavak}
```

```

9975 \providetranslation{Notation (glossaries)}{Kifejez\`es}
9976 \providetranslation{Description (glossaries)}{Magyar\`azat}
9977 \providetranslation{Symbol (glossaries)}{Jel\`ol\`es}
9978 \providetranslation{Page List (glossaries)}{Oldalsz\`am}
9979 \providetranslation{Symbols (glossaries)}{Jelek}
9980 \providetranslation{Numbers (glossaries)}{Sz\`amjegyek}

```

7.12 Polish Dictionary

This is a dictionary file provided for use with the package.

```

9981 \ProvidesDictionary{glossaries-dictionary}{Polish}

```

Provide Polish translations:

```

9982 \providetranslation{Glossary}{S{\l}ownik termin\`ow}
9983 \providetranslation{Acronyms}{Skr\`ot}
9984 \providetranslation{Notation (glossaries)}{Termin}
9985 \providetranslation{Description (glossaries)}{Opis}
9986 \providetranslation{Symbol (glossaries)}{Symbol}
9987 \providetranslation{Page List (glossaries)}{Strony}
9988 \providetranslation{Symbols (glossaries)}{Symbole}
9989 \providetranslation{Numbers (glossaries)}{Liczby}

```

7.13 Serbian Dictionary

This dictionary was provided by Zoran Filipovic.

```

9990 \ProvidesDictionary{glossaries-dictionary}{Serbian}
9991 \providetranslation{Glossary}{Mali re\`v cnik}
9992 \providetranslation{Acronyms}{Skra\` cenice}
9993 \providetranslation{Notation (glossaries)}{Oznaka}
9994 \providetranslation{Description (glossaries)}{Opis}
9995 \providetranslation{Symbol (glossaries)}{Simbol}
9996 \providetranslation{Page List (glossaries)}{Stranica}
9997 \providetranslation{Symbols (glossaries)}{Simboli}
9998 \providetranslation{Numbers (glossaries)}{Brojevi}

```

7.14 Spanish Dictionary

This is a dictionary file provided for use with the package.

```

9999 \ProvidesDictionary{glossaries-dictionary}{Spanish}

```

Provide Spanish translations:

```

10000 \providetranslation{Glossary}{Glosario}
10001 \providetranslation{Acronyms}{Siglas}
10002 \providetranslation{Notation (glossaries)}{Entrada}
10003 \providetranslation{Description (glossaries)}{Descripci\`on}
10004 \providetranslation{Symbol (glossaries)}{S\`{\i}mbolo}
10005 \providetranslation{Page List (glossaries)}{Lista de p\`aginas}
10006 \providetranslation{Symbols (glossaries)}{S\`{\i}mbolos}
10007 \providetranslation{Numbers (glossaries)}{N\`umeros}

```

Glossary

`makeindex` An indexing application. 8, 19

`xindy` An flexible indexing application with multilingual support written in Perl. 8, 19

Change History

- 1.01
 - General: Added range facility in format key 86
 - `\writeist`: Added spaces after `\delimN` and `\delimR` in ist file 143
- 1.03
 - `\makefirstuc`: changed 'protected@edef to 'def 223
- 1.04
 - General: Added `\glstextformat` 72
- 1.05
 - `\glossarysection`: added `\@mkboth to \glossarysection` 31
 - `\gls@defglossaryentry`: Changed the default value of the sort key to just the value of the name key 62
 - `\glsmakefirstuc`: new 224
- 1.06
 - General: now requires `etoolbox` . 222
 - `\capitalisewords`: new 224
 - `\xcapitalisewords`: new 224
- 1.07
 - `\@gls@link`: fixed bug caused by `\theglsentrycounter` setting the page number too soon 84
 - `\glsadd`: fixed bug caused by `\theglsentrycounter` setting the page number too soon 140
- 1.08
 - General: Added babel support ... 26
 - `\capitalisewords`: made robust 224
- `listgroup`: changed `listgroup` style to use `\glsgetgrouptitle` 230
 - `altlistgroup`: changed `altlistgroup` style to use `\glsgetgrouptitle` 231
 - `\makefirstuc`: made robust ... 223
- 1.1
 - `\@glossarysection`: numbered sections and auto label added 32
 - `\@gls@tmpb`: changed `\toksdef to \newtoks` 88
 - `\@gls@toc`: numberline added .. 34
 - `\@p@glossarysection`: numbered sections and auto label added 33
 - General: Added support for translator package 26
 - `amsgen` now loaded (`\new@ifnextchar` needed) 2
 - `translate`: `translate` option added 17
 - `\setglossarysection`: new ... 32
 - `numberedsection`: numbered-section package option added . 4
 - `numberline`: `numberline` option added 4
- 1.12
 - `\@GLSpl`: now uses `\glsentrydescplural` and `\glsentrysymbolplural` instead of `\glsentrydesc` and `\glsentrysymbol` 102
 - `\@Glspl`: now uses `\glsentrydescplural` and `\glsentrysymbolplural` instead of `\glsentrydesc` and `\glsentrysymbol` 101

General: added check for \hypertarget separate to \hyperlink (memoir de- fines \hyperlink but not \hypertarget) 94	\glsautoprefix:new 4
descriptionplural:new 50	\glsnavhyperlink: changed 'edef to 'protected@edef ... 225
\gls@defglossaryentry: Changed default first plural to be first key with s appended (was text key with s appended) 62	\glsnavhypertarget: added write to aux file 225
descriptionplural support added 62	\glsnavigation: changed to only use labels for groups that are present 226
symbolplural support added .. 62	1.15
\Glsentrydescplural:New .. 135	\@gls@link: added \glslabel . 84
\glsentrydescplural:New .. 134	General: Added \glssettoctitle 26
\Glsentrysymbolplural:New 135	\gls@defglossaryentry: check for \@glo@first in descrip- tion 66
\glsentrysymbolplural:New 135	check for \@glo@text in sym- bol 66
\SetDescriptionFootnoteAcronymStyle: Added \protect before \footnote and \glslink . 196	\gls@hypergroup: new . 226
\SetFootnoteAcronymStyle: Added \protect before \footnote and \glslink . 203	\glsnavhypertarget: added check if rerun required 225
symbolplural: new 51	\glssettoctitle: new 25
1.13	\printglossary: changed the way the TOC title is set 158
General: Add Polish support 315, 318 fixed bug that ignored 3rd pa- rameter 104–117	1.16
\ACRfullpl:new 179	\@GLS@: Test glossary type is \acronymtype in addition to checking if footnote option has been used 98
\Acrfullpl:new 178	\@GLSpl: Test glossary type is \acronymtype in addition to checking if footnote option has been used 102
\acrfullpl:new 178	\@GLs@: Test glossary type is \acronymtype in addition to checking if footnote option has been used 97
\acrpluralsuffix:New 175	\@GLspl@: Test glossary type is \acronymtype in addition to checking if footnote option has been used 101
\gls@defglossaryentry: Changed default first value .. 62	\@gls@: Test glossary type is \acronymtype in addition to checking if footnote option has been used 96
Changed default firstplural value 62	\@glsdisp: Test glossary type is \acronymtype in addition to checking if footnote option has been used 103
Removed restriction on only using \newglossaryentry in the preamble 67	
\newacronym: Removed re- striction on only using \newacronym in the preamble 175	
1.14	
\@gls@hypergroup: new 226	
General: added nonnumberlist key to \printglossary 162	
added numberedsection key to \printglossary 162	
\firstacronymfont: new 179	

\@glspl@: Test glossary type is \acronymtype in addition to checking if footnote option has been used	99	\ifglxindy: new	19
\@glstarget: raised the hyper- target so the target text doesn't scroll off the top of the page .	94	\istfilename: added xindy sup- port	28
\gls@defglossaryentry: Changed def to let	62	\newglossarystyle: made \newglossarystyle long .	171
1.17		\nopostdesc: new	27
\@do@wrglossary: new	153	nonumberlist: new	52
\@do@seeglossary: new	156	\printglossary: added check to determine if \printglossary is already defined	158
\@glo@storeentry: new	68	added print language to aux file	158
\@glossary: changed defini- tion to use \index instead of \@index	152	order: order package option added	19
\@glsdefaultplural: new	54	\writeist: added xindy support	143
\@glsdefaultsort: new	54	1.18	
\@glshypernumber: new	172	\@gls@loadlist: new	6
\@glsnoname: new	53	\@gls@loadlong: new	6
\@glsnonextpages: new	162	\@gls@loadsuper: new	6
\@wrglossary: modified to allow for xindy support	152	\@gls@loadtree: new	6
General: added Brazilian dictio- nary	319	\gls@defglossaryentry: Changed default value of sort to \@glsdefaultsort	62
Added Brazilian support	315	moved sort sanitization to \newglossaryentry	66
added xindy support	19	\glstarget: new	165
parent: new	52	\oldacronym: new	174
see: new	52	nolist: new	6
\gls@defglossaryentry: added nonumberlist key	63	nolong: new	6
added parent key	63	sort: moved sanitization to \newglossaryentry	50
added see key	62	nostyles: new	7
Stored main part of entry format when entry is defined	67	nosuper: new	6
\gls@suffixF: new	29	notree: new	7
\gls@suffixFF: new	29	1.19	
\glshyperlink: new	140	\glsclearpage: new	34
\glshypernumber: modified to allow material to be attached to location	172	\glsdisp: new	103
\glsnavhyperlink: replaced 'hy- perlink to '@glslink	225	\SetDescriptionAcronymStyle: changed \acronymfont to use \textsmaller instead of \smaller	201
\glsnavhypertarget: replaced 'hypertarget to '@glstarget .	225	\SetDescriptionFootnoteAcronymStyle: changed \acronymfont to use \textsmaller instead of \smaller	197
\glssee: new	156	\SetFootnoteAcronymStyle: changed \acronymfont to use \textsmaller instead of \smaller	203
\glsseeformat: new	156		
\glsSetSuffixF: new	29		
\glsSetSuffixFF: new	29		

\SetSmallAcronymStyle:		\@gls@link: added \leavevmode	
changed \acronymfont to	 84	
use \textsmaller instead of		Moved entry existence check to	
\smaller		avoid duplicate code	
206		84	
1.2	General: fixed bug in ngerman	\@glsdisp: Added check for hy-	
	captions	perfirst	103
312		\@glspl@: Added check for hyper-	
2.01		first	99
	\@gls@link: moved \@do@wrglossary	\gls glossarymark: Added check	
	before term is displayed to pre-	to see if it's already defined ..	31
	vent unwanted whatsit	hyperfirst: new	18
	84	2.04	
	\forall glossaries: replaced	\@GLS@: Changed test to check if	
	\ifthenelse with \ifx	glossary type has been identi-	
	43	fied as a list of acronyms	98
	\forall gloss entries: replaced	\@GLSpl@: Changed test to check if	
	\ifthenelse with \ifx	glossary type has been identi-	
	43	fied as a list of acronyms ...	102
	\glsdefmain: new	\@Gls@: Changed test to check if	
	10	glossary type has been identi-	
	\glsdescwidth: changed	fied as a list of acronyms	97
	\linewidth to \hsize . 233, 248	\@Glspl@: Changed test to check	
	\glslistdottedwidth: changed	if glossary type has been iden-	
	\linewidth to \hsize	tified as a list of acronyms ..	101
	232	\@glossaryentryfield: new ..	67
	\gls pagelistwidth: changed	\@glossarysubentryfield:	
	\linewidth to \hsize . 233, 248	new	67
	nomain: added nomain package	\@gls@: Changed test to check if	
	option	glossary type has been identi-	
	11	fied as a list of acronyms	96
	\writeist: removed item_02 - no	\@glsacronymlists: new	12
	such makeindex key	\@glsdisp: Changed test to check	
	147	if glossary type has been iden-	
	2.02	tified as a list of acronyms ..	103
	General: Changed Brazil to Brazil-	\@glspl@: Changed test to check	
	ian	if glossary type has been iden-	
	319	tified as a list of acronyms ...	99
	false will prevent automatic	\@newglossaryentryposthook:	
	loading of translator package	new	67
	23	\@newglossaryentryprehook:	
	\glossarysection: changed	new	67
	\@mkboth to \glossarymark	acronymlists: new	13
	31	\DeclareAcronymList: new ...	12
	\gls glossarymark: New	\DefineAcronymSynonyms: new	191
	31	\gls@def glossaryentry: added	
	\printglossary: suppressed	user1-6 keys	63
	warning globally rather than	\glsadd: fixed bug that ignored	
	locally	counter	140
	161	\Glsentryuseri: new	136
2.03			
	\@GLS@: Added check for hyper-		
	first		
	98		
	\@GLSpl@: Added check for hyper-		
	first		
	102		
	\@Gls@: Added check for hyper-		
	first		
	97		
	\@Glspl@: Added check for hyper-		
	first		
	101		
	\@gls@: Added check for hyper-		
	first		
	96		
	\@gls@link: new		
	84		

\glsentryuseri:new	136	2.07	
\Glsentryuserii:new	137		General: glssadd format key
\glsentryuserii:new	136		stored in \@glsnumberformat
\Glsentryuseriii:new	137		(was mistakenly stored in
\glsentryuseriii:new	137		\@glo@format) 140
\Glsentryuseriv:new	137	3.0	
\glsentryuseriv:new	137		\@@do@wrglossary: added check
\Glsentryuseriv:new	137		for hyper location prefix ... 154
\Glsentryuseriv:new	137		modified to use new format .. 153
\glsentryuseriv:new	137		\@@glossarysec: replaced
\Glsentryuseriv:new	137		\@ifundefined with
\glsentryuseriv:new	137		\ifcsundef 4
\newglossary: added check to			\@do@seeglossary: Sanitize and
determine if \gls@<type>@display			escape cross-referencing in-
and \gls@<type>@displayfirst			formation 156
have been defined. 48			\@gls@counterwithin: new 8
\SetAcronymLists:new	13		\@gls@ifinlist:new 35
\SetDefaultAcronymDisplayStyle:			\@gls@link: added \@gls@saveentrycounter
new 193		 84
\SetDefaultAcronymStyle:			added \@gls@setsort 84
new 194			\@gls@saveentrycounter: new 85
\SetDescriptionAcronymDisplayStyle:			\@gls@setupsort@def: new 9
new 199			\@gls@setupsort@standard:
\SetDescriptionDUAAcronymDisplayStyle:			new 8
new 197			\@gls@setupsort@use: new 9
\SetDescriptionFootnoteAcronymDisplayStyle:			\@gls@xdy@locationlist: new 38
new 195			\@gls@link: replaced \@ifundefined
\SetDUADisplayStyle: new .. 206			with \ifcsundef 94
\SetFootnoteAcronymDisplayStyle:			\@gls@nextpages: new 162
new 201			\@makeglossary: Added check
\SetSmallAcronymDisplayStyle:			for savewrites 149
new 204			\@set@glo@numformat: added
2.05			4th argument 86
\@glsdisp: Added closing brace.			\@wrglossary: modified to take
Patch provided by Sergiu			into account savewrites 152
Dotenco 103			\@xdy@attributelist: new 34
Removed spurious brace. Patch			General: added prefix to hyperlink
provided by Sergiu Dotenco 104		 173
\writeist: Added \string be-			etoolbox now loaded 2
fore opening and closing			replaced \@ifundefined with
braces. Patch provided by			\ifcsundef 24,82,161
Segiu Dotenco 147			\acrfootnote: new 194
2.06			\ACRfull: added starred version 177
\altnewglossary: new 48			\Acrfull: added starred version 177
\CustomAcronymFields:new . 209			\acrfull: added starred version 176
\CustomNewAcronymDef:new . 209			\ACRfullpl: added starred ver-
\SetCustomDisplayStyle:new 209			sion 179
\SetCustomStyle:new 209			\Acrfullpl: added starred ver-
			sion 178

<code>\acrfullpl</code> : added starred version	178	<code>\glentrytext</code>	140
<code>\acrlinkfootnote</code> : new	194	<code>\glshypernumber</code> : replaced	
<code>\acrnolinkfootnote</code> : new ...	195	<code>\@ifundefined</code> with	
<code>\addglossarytocaptions</code> : replaced		<code>\ifcsundef</code>	172
<code>\@ifundefined</code> with		<code>\glsnumberformat</code> : replaced	
<code>\ifcsundef</code>	26	<code>\@ifundefined</code> with	
<code>savewrites</code> : new	20	<code>\ifcsundef</code>	30
<code>see</code> : added <code>\@gls@seeautonumberlist</code>		<code>\glsrefentry</code> : new	164
.....	52	<code>\glsresetsubentrycounter</code> :	
<code>seeautonumberlist</code> : new	6	new	163
<code>\glossarysection</code> : replaced		<code>\glsseeitem</code> : hyperlink uses	
<code>\@ifundefined</code> with		<code>\glsseeitemformat</code> instead	
<code>\ifcsundef</code>	31	of <code>\glentryname</code>	157
<code>\glossarystyle</code> : replaced		<code>\glsseeitemformat</code> : new	157
<code>\@ifundefined</code> with		<code>\glsortnumberfmt</code> : new	9
<code>\ifcsundef</code>	170	<code>\glsstepentry</code> : new	164
<code>\gls@codepage</code> : replaced		<code>\glsstepsubentry</code> : new	164
<code>\@ifundefined</code> with		<code>\glsesubentrycounterlabel</code> :	
<code>\ifcsundef</code>	20	new	165
<code>\gls@defglossaryentry</code> : added		<code>\glsesubentryitem</code> : new	165
<code>\@gls@defsort</code>	66	<code>theglossary</code> : replaced <code>\@ifundefined</code>	
added short and long keys	63	with <code>\ifcsundef</code>	165
replaced <code>\@ifundefined</code> with		<code>short</code> : new	53
<code>\ifcsundef</code>	63	<code>shortplural</code> : new	53
<code>\gls@doclearpage</code> : replaced		<code>\ifglossaryexists</code> : re-	
<code>\@ifundefined</code> with		placed <code>\@ifundefined</code> with	
<code>\ifcsundef</code>	33	<code>\ifcsundef</code>	44
<code>\glsadd</code> : added <code>\@gls@saveentrycounter</code>		<code>\ifglentryexists</code> : re-	
.....	141	placed <code>\@ifundefined</code> with	
<code>\GlsAddXdyCounters</code> : new	35	<code>\ifcsundef</code>	44
<code>\glentrycounterlabel</code> : new	164	<code>\istfile</code> : deprecated	151
<code>\glentryitem</code> : new	165	<code>glossaryentry</code> : new	163
<code>\Glsentrylong</code> : new	138	<code>glossarysubentry</code> : new	163
<code>\glentrylong</code> : new	138	<code>\newglossary</code> : added <code>\@gls@defsortcount</code>	
<code>\Glsentrylongpl</code> : new	138	48
<code>\glentrylongpl</code> : new	138	replaced <code>\@ifundefined</code> with	
<code>\Glsentryshort</code> : new	138	<code>\ifcsundef</code>	48
<code>\glentryshort</code> : new	137	<code>\newglossaryentry</code> : replaced	
<code>\Glsentryshortpl</code> : new	138	<code>\DeclareRobustCommand</code>	
<code>\glentryshortpl</code> : new	138	with <code>\newrobustcmd</code>	56
<code>\glsgetgrouptitle</code> : re-		<code>\newglossarystyle</code> : re-	
placed <code>\@ifundefined</code> with		placed <code>\@ifundefined</code> with	
<code>\ifcsundef</code>	169	<code>\ifcsundef</code>	171
<code>\glsglossarymark</code> : replaced		<code>entrycounter</code> : new	7
<code>\@ifundefined</code> with		<code>entrycounterwithin</code> : new	8
<code>\ifcsundef</code>	31	<code>\oldacronym</code> : replaced <code>\@ifundefined</code>	
<code>\glshyperlink</code> : changed de-		with <code>\ifcsundef</code>	174
fault from <code>\glentryname</code> to		<code>compatible-2.07</code> : compatible-	
		2.07 option added	21

long:new	53		
longplural:new	53		
nonumberlist:now boolean ...	52		
sort:new	8		
counter:replaced \@ifundefined with \ifcsundef	51	3.01	
\printglossary: added			
\currentglossary	159		
added \glsnextpages	159		
make toctitle default to title ..	159		
replaced \@ifundefined with \ifcsundef	158, 160		
\SetDescriptionFootnoteAcronymDisplayStyle:			
expanded options link op- tions	195		
\setentrycounter: added op- tional argument	170		
\showacronymlists:new	214		
\showglocounter:new	212		
\showglodesc:new	213		
\showglodescplural:new ...	213		
\showglofirst:new	211		
\showglofirstpl:new	211		
\showgloflag:new	214		
\showgloindex:new	214		
\showglolevel:new	211		
\showglongame:new	213		
\showgloparent:new	211		
\showgloplural:new	211		
\showglosort:new	213		
\showglossaries:new	214		
\showglossarycounter:new .	215		
\showglossaryentries:new .	215		
\showglossaryin:new	214		
\showglossaryout:new	215		
\showglossarytitle:new ...	215		
\showglosymbol:new	213		
\showglosymbolplural:new .	213		
\showglotext:new	211		
\showglotype:new	211		
\showglouserii:new	212		
\showglouseriii:new	212		
\showglouseriv:new	212		
\showglouservi:new	213		
subentrycounter:new	8		
			\writeist: added xindy-only macro definitions to glossary open tag 145 modified to support new for- mat 143
			\@glswritefiles: added check for empty glossaries 151
			General: made robust 97
			\ACRfull: made robust 177
			\Acrfull: made robust 177
			\acrfull: made robust 176
			\acrfullformat: removed \acronymfont as it should al- ready be set in the second ar- gument. 177
			\ACRfullpl: made robust 179
			\Acrfullpl: made robust 178
			\acrfullpl: made robust 178
			\ACRlong: made robust 131
			\Acrlong: made robust 130
			\acrlong: made robust 129
			\ACRlongpl: made robust 133
			\Acrlongpl: made robust 132
			\acrlongpl: made robust 131
			\ACRshort: made robust 127
			\Acrshort: made robust 126
			\acrshort: made robust 125
			\ACRshortpl: made robust 129
			\Acrshortpl: made robust 128
			\acrshortpl: made robust 127
			\Gls: made robust 96
			\glsadd: made robust 140
			\glsaddall: made robust 141
			\GLSdesc: made robust 112
			\Glsdesc: made robust 111
			\glsdesc: made robust 111
			\GLSdescplural: made robust . 113
			\Glsdescplural: made robust . 113
			\glsdescplural: made robust . 112
			\glsfirst: made robust 105
			\GLSfirstplural: made robust 109
			\Glsfirstplural: made robust 109
			\glsfirstplural: made robust 108
			\glslink: made robust 83
			\GLSname: made robust 110
			\Glsname: made robust 110
			\glsname: made robust 109
			\GLSpl: made robust 101

\Glspl: made robust	100	General: added check for polyglos-	
\glspl: made robust	99	sia	23
\GLSplural: made robust	108	reversed order of package check	27
\GLSsymbol: made robust	115	savenumberlist: new	6
\Glsymbol: made robust	114	ucmark: new	7
\glssymbol: made robust	114	\gls@defglossaryentry: added	
\GLSsymbolplural: made robust		numberlist element	66
.....	116	\gls@save@numberlist: new ..	158
\Glsymbolplural: made robust		\glsdisplaynumberlist: new	139
.....	116	\glsentrycounter: set default	
\glssymbolplural: made robust		value	85
.....	115	\Glsentryfull: fixed bug (re-	
\Glstext: made robust	105	placed \glsentryshortpl	
\glstext: made robust	104	with \glsentryshort)	138
\GLSuseri: made robust	118	\glsentryfullpl: fixed bug (re-	
\Glsuseri: made robust	117	placed \glsentryshort with	
\gluseri: made robust	117	\glsentryshortpl)	138
\GLSuserii: made robust	119	\glsentrynumberlist: new ..	139
\Glsuserii: made robust	119	\glsmoveentry: new	67
\gluserii: made robust	118	\glsnumlistlastsep: new ...	140
\GLSuseriii: made robust	120	\glsnumlistsep: new	140
\Glsuseriii: made robust	120	\glsresetsubentrycounter:	
\gluseriii: made robust	120	new	164
\GLSuseriv: made robust	122	\ifglshaschildren: new	45
\Glsuseriv: made robust	121	\ifglshasparent: new	45
\gluseriv: made robust	121	\makeglossaries: added list	
\GLSuserv: made robust	123	parser	150
\Glsuserv: made robust	123	indexonlyfirst: new	18
\gluserv: made robust	122	\printglossary: add a way to	
\GLSuservi: made robust	125	fetch current entry label ...	159
\Glsuservi: made robust	124	\renewglossarystyle: new ..	171
\gluservi: made robust	124	\showglossaryentries: fixed	
		misspelt command	215
3.02		\SmallNewAcronymDef: fixed	
\@do@wrglossary: changed		broken short and long plural	204
\@glslocref to \theglsentrycounter			
.....	155	3.03	
\@do@wrglossary: changed		\@gls@sanitizesort: new	16
\@do@wr@glossary to test for		\@gls@setupsort@standard:	
indexonlyfirst option; put old		used \@gls@sanitizesort ..	9
\@do@wr@glossary code into		General: allow title to set toctitle	161
\@do@wrglossary	153	\glsinlinedescformat: new ..	229
\@gls@missingnumberlist:		\glsinlineemptydescformat:	
new	54	new	229
\@glswritefiles: added check		\glsinlinenameformat: new ..	229
for existence of token in case		\glsinlinepostchild: new ..	229
\makeglossaries has been		\glsinlinesubdescformat:	
omitted	151	new	229
\@wrglossary: added check for		\glsinlinesubnameformat:	
glossary file defined	152	new	229

<code>\glspostinline:</code> replaced “.” with <code>\glspostdescription</code> .. 229	General: Added check for doc package 2
<code>altnogragged4col:</code> added check for <code>glsnogroupskip</code> .. 243	added <code>datatool-base</code> as a re- quired package 2
<code>altsuperragged4col:</code> added check for <code>glsnogroupskip</code> .. 259	added local key 83
<code>almtree:</code> added check for <code>glsnogroupskip</code> 267	<code>\gls@Alphpage:new</code> 153
<code>index:</code> added check for <code>glsnogroupskip</code> 261	<code>\gls@alphpage:new</code> 153
<code>nogroupskip:new</code> 7	<code>\gls@disablepagerefexpansion:</code> new 153
<code>long:</code> added check for <code>glsnogroupskip</code> 233	<code>\gls@numberpage:new</code> 153
<code>long3col:</code> added check for <code>glsnogroupskip</code> 235	<code>\gls@protected@pagefmts:</code> new 153
<code>long4col:</code> added check for <code>glsnogroupskip</code> 236	<code>\gls@romanpage:new</code> 153
<code>longragged:</code> added check for <code>glsnogroupskip</code> 240	<code>\glsdefmain:</code> added check for doc package 10
<code>longragged3col:</code> added check for <code>glsnogroupskip</code> 241	<code>\glsorg@endtheglossary:new</code> . 3
<code>nopostdot:new</code> 7	<code>\glsorg@glossary:new</code> 2
<code>\printglossary:</code> allow title to override default toctitle 158	<code>\glsorg@theglossary:new</code> 3
<code>tree:</code> added check for <code>glsnogroupskip</code> 263	<code>\glsorg@wrglossary:new</code> 3
<code>treename:</code> added check for <code>glsnogroupskip</code> 264	<code>altlist:</code> replaced <code>\newline</code> with paragraph break 231
<code>super:</code> added check for <code>glsnogroupskip</code> 249	<code>\PrintChanges:new</code> 3
<code>super3col:</code> added check for <code>glsnogroupskip</code> 251	<code>\printglossary:</code> Moved aux write to end of document to prevent unwanted whatsit oc- curring here. 160
<code>super4col:</code> added check for <code>glsnogroupskip</code> 252	
<code>superragged:</code> added check for <code>glsnogroupskip</code> 256	
<code>superragged3col:</code> added check for <code>glsnogroupskip</code> 257	
3.04	3.05
<code>\@@do@wrglossary:</code> changed <code>\theglstrycounter</code> back to <code>\@glslocref</code> 155	<code>\@@do@wrglossary:</code> add Roman case. Fixed bugs in the else statements 154
modified to compensate for possible incorrect page num- ber 154	<code>\@gls@link:</code> added check for “no- hypertypes” 84
<code>\@gls@escbsdq:</code> unsani- tize <code>\gls@numberpage</code> , <code>\gls@alphpage</code> , <code>\gls@Alphpage</code> and <code>\gls@romanpage</code> 87	<code>\@gls@nohyperlist:new</code> 14
	<code>mcolalmtree:</code> replaced ‘2’ with <code>\glsmcols</code> 247
	<code>mcolindex:</code> replaced ‘2’ with <code>\glsmcols</code> 245
	<code>mcoltree:</code> replaced ‘2’ with <code>\glsmcols</code> 245
	<code>mcoltreename:</code> replaced ‘2’ with <code>\glsmcols</code> 246
	<code>\gls@protected@pagefmts:</code> added Roman to list 153
	<code>\gls@Romanpage:new</code> 153
	<code>\GlsDeclareNoHyperList:new</code> 14
	<code>\glsgetgrouplabel:</code> fixed bug (typo in <code>\equal</code>) 170
	<code>\nopostdesc:</code> made robust 27
	<code>nohypertypes:new</code> 14

3.06	\@xdy@main@language: Changed back to using \language name 20	\gls@assign@descplural@field: new 15
	\findrootlanguage: Obsoleted 41	\gls@assign@field: new 56
3.07		\gls@ifnotmeasuring: new ... 69
	\@gls@link: fixed bug that failed to find entry in list 84	\glsaddallunused: new 141
	\glossarypreamble: modified to work with \setglossarypreamble 30	\glsexpandfields: new 56
	\gls@doclearpage: added check for openright 33	\glsnoexpandfields: new 56
	\glspostdescription: Added spacefactor code 7	\glssee: made robust 156
	\GlsSetXdyCodePage: Added check for fontspec 42	\glsseeformat: made robust .. 156
	\SetDescriptionAcronymDisplayStyle: now using \glsdoparenifnotempty 199	\glsseeitem: made robust 157
	\setglossarypreamble: new .. 30	\glsseelist: made robust 157
3.08a		\ifglsgdescsuppressed: new .. 45
	\@glo@storeentry: no longer need to check for special char- acters in any of the fields other than sort 68	\ifglshasdesc: new 45
	updated for \glossentry 68	\ifglshassymbol: new 46
	\@glossaryentryfield: switched to \glossentry 67	list: updated list style to use \glossentry and \subglossentry 230
	\@glossarysubentryfield: switched to \subglossentry 67	listdotted: updated listdotted style to use \glossentry and \subglossentry 232
	General: added nogroupskip key to \printglossary 162	altlist: updated altlist style to use \glossentry and \subglossentry 231
	removed definition of \@glossaryentryfield .. 303	altlongragged4col: updated to use \glossentry and \subglossentry 242
	removed definition of \@glossarysubentryfield 303	alttree: updated to use \glossentry and \subglossentry 265
	\compatibleglossentry: new 166	index: added paragraph break at end of environment 261
	\compatiblesubglossentry: new 167	updated to use \glossentry and \subglossentry 261
	\glossaryentryfield: depre- cated 168	inline: updated inline style to use \glossentry and \subglossentry 227
	\Glossentrydesc: new 167	long: updated to use \glossentry and \subglossentry 233
	\glossentrydesc: new 166	longragged: updated to use \glossentry and \subglossentry 239
	\Glossentryname: new 166	longragged3col: updated to use \glossentry and \subglossentry 241
	\glossentryname: new 166	tree: updated to use \glossentry and \subglossentry 262
	\Glossentrysymbol: new 167	\setglossarystyle: new 170
	\glossentrysymbol: new 167	\setglossentrycompatibility: new 167
	\gls@assign@desc@field: new 15	

superragged: updated to	new	62
use \glossentry and	\glswritedefhook: new	61
\subglossentry	\makeglossaries: Added	
3.09a	providecommand code to aux	
\@gls@assign@symbolplural@field:	file	149
new	\new@glossaryentry: new	56
\@gls@default@value: new ...	\newglossary: added \@gls@provide@newglossary	
\Glsentrydesc: made robust	48
\Glsentrydescplural: made ro-	\printglossary: Added provide-	
bust	command code to aux file ..	160
\Glsentryfirst: made robust .	3.11a	
\Glsentryfirstplural: made	\@ACRlong: added \glslabel,	
robust	\glsifplural, \glscapscase,	
\Glsentryfull: made robust ..	\glsinsert and \glscustomtext	303
\Glsentryfullpl: made robust	\@ACRshort: added \glslabel,	
\Glsentrylong: made robust ..	\glsifplural, \glscapscase,	
\Glsentrylongpl: made robust	\glsinsert and \glscustomtext	302
\Glsentryname: made robust ..	\@Acrlong: added \glslabel,	
\Glsentryplural: made robust	\glsifplural, \glscapscase,	
\Glsentryshort: made robust .	\glsinsert and \glscustomtext	303
\Glsentryshortpl: made robust	301
.....	\@GLS@: add \glslabel,	
\Glsentrysymbol: made robust	\glsifplural, \glscapscase,	
\Glsentrysymbolplural: made	\glscustomtext and	
robust	\glsinsert	98
\Glsentrytext: made robust ..	change to using \glsentryfmt	
\Glsentryuseri: made robust .	style commands	98
\Glsentryuserii: made robust	removed \MakeUppercase	
\Glsentryuseriii: made robust	(now moved to \glsentryfmt)	
.....	98
\Glsentryuseriv: made robust	\@GLSpl: add \glslabel,	
\Glsentryuserv: made robust .	\glsifplural, \glscapscase,	
\Glsentryuservi: made robust	\glscustomtext and	
\glstextup: new	\glsinsert	102
\if@gls@docloaded: Add a fix for	change to using \glsentryfmt	
\RecordChanges	style commands	102
\ifglshassymbol: changed test	removed \MakeUppercase	
to check for \@gls@default@symbol	as now dealt with in	
.....	\glsentryfmt	102
3.10a	\@Gls@: add \glsifplural,	
\@gls@keymap: new	\glscapscase, \glscustomtext	
\@gls@provide@newglossary:	and \glsinsert	97
new		
\@glsdefaultplural: Obsolete .		
\@glsnodesc: new		
\gls@assign@type@field: new		
15		
\gls@defglossaryentry:		
Changed to using \@gls@default@value		
.....		
62		

change to using \glentryfmt style commands	97	changed to just use \glentrydesc	111, 112
removed \makefirstuc (now dealt with in \glentryfmt)	97	changed to just use \Glsentryfirstplural	109
\@Glspl@: add \glsifplural, \glscapscase, \glscustomtext and \glsinsert	101	changed to just use \glentryfirstplural	108, 109
change to using \glentryfmt style commands	101	changed to just use \Glsentryfirst	106
removed \makefirstuc (now dealt with in \glentryfmt)	101	changed to just use \glentryfirst	106, 107
\@acrlong: added \glslabel, \glsifplural, \glscapscase, \glsinsert and \glscustomtext	302	changed to just use \Glsentryname	110
\@acrshort: added \glslabel, \glsifplural, \glscapscase, \glsinsert and \glscustomtext	301	changed to just use \glentryname	110, 111
\@gls@: add \glslabel, \glsifplural, \glscapscase, \glscustomtext and \glsinsert	95	changed to just use \Glsentryplural	107
change to using \glentryfmt style commands	96	changed to just use \glentryplural	107, 108
\@gls@noexpand@fields: Fixed bug expand replaced with noexpand	55	changed to just use \Glsentrysymbolplural	116
\@glsdisp: add \glslabel, \glsifplural, \glscapscase, \glscustomtext and \glsinsert	103	changed to just use \glentrysymbolplural	116, 117
change to using \glentryfmt style commands	103	changed to just use \Glsentrysymbol	115
\@Glspl@: add \glslabel, \glsifplural, \glscapscase, \glscustomtext and \glsinsert	99	changed to just use \glentrysymbol	114, 115
General: added \glslabel, \glsifplural, \glscapscase, \glsinsert and \glscustomtext	126–133	Changed to just use \Glsentrytext	105
changed to just use \Glsentrydescplural	113	changed to just use \glentrytext	104
changed to just use \glentrydescplural	113, 114	changed to just use \Glsentryuseriii	120
changed to just use \Glsentrydesc	112	changed to just use \glentryuseriii	120, 121
		changed to just use \Glsentryuserii	119
		changed to just use \glentryuserii	118, 119
		changed to just use \Glsentryuseriv	122
		changed to just use \glentryuseriv	122
		changed to just use \Glsentryuseri	118
		changed to just use \glentryuseri	117, 118
		changed to just use \Glsentryuservi	125

changed to just use \glentryuservi	209
..... 124, 125	
changed to just use \Glsentryuserv	
..... 123	
changed to just use \glentryuserv	
..... 123, 124	
Now requires textcase	2
acronymlists: replaced	
\@addtoacronymlists with	
\DeclareAcronymList	13
\defglstdisplay: obsoleted	81
\defglstdisplayfirst: obso-	
leted	82
\defglsentryfmt: new	47
\forglsentries: replaced \ifx	
with \ifdefempty	43
\gls@assign@desc: new	61
\gls@defglossaryentry: Fixed	
default counter if none sup-	
plied	65
\gls@doentryfmt: new	47
\glsdisplay: obsoleted	81
\glsdisplayfirst: obsoleted	81
\glsgenentryfmt: new	76
\glsgetgrouptitle: Added	
check in case non-Latin alpha-	
bet in use	169
\glsglossarymark: replaced	
\MakeUppercase with	
\mfirstucMakeUppercase	31
\glsnavigation: switched to us-	
ing \@gls@getgrouptitle	226
\ifglshasdesc: replaced	
\ifdefempty with \ifcsemt	
y	45
\ifglshaslong: new	46
\ifglshasshort: new	46
\ifglshassymbol: replaced	
\ifdefempty with \ifcsemt	
y	46
\ifglused: replaced \ifthenelse	
with \ifbool	44
\longnewglossaryentry: new	61
\newglossary: replaced	
\glsdisplay and \glsdisplayfirst	
with \glentryfmt	48
compatible-3.07: cnew	21
\SetCustomDisplayStyle: up-	
dated to use \defglsentryfmt	
\SetDefaultAcronymDisplayStyle:	
changed to use \defglsentryfmt	193
\SetDescriptionAcronymDisplayStyle:	
updated to use \defglsentryfmt	199
\SetDescriptionDUAAcronymDisplayStyle:	
updated to use \defglsentryfmt	197
\SetDescriptionFootnoteAcronymDisplayStyle:	
updated to use \defglsentryfmt	195
\SetDUADisplayStyle: updated	
to use \defglsentryfmt	206
\SetFootnoteAcronymDisplayStyle:	
updated to use \defglsentryfmt	201
\SetSmallAcronymDisplayStyle:	
updated to use \defglsentryfmt	204
\setupglossaries: new	22
\showglolong: new	214
\showgloshort: new	214
numbers: new	21
symbols: new	21
3.12a	
\gls@defglossaryentry: added	
\glslabel	62
\glsaddkey: new	58
3.13a	
\@gls@assign@symbol@field:	
changed to use \glsetnoexpandfield	15
\@gls@assign@symbolplural@field:	
changed to use \glsetnoexpandfield	15
\@gls@link: removed \relax	85
\@gls@notranslatorhook: new	17
\@gls@setupsort@standard:	
moved \@gls@santizesort	
to \glsprestandardsort	9
General: added cs@gls@notranslatorhook	
to else clause	27
ucmark: added check for memoir	7
see: added \gls@checkseeallowed	
.....	52
\glossarysection: changed	
\glossarymark to \glsglossarymark	

..... 31	longheader: switched to
\glossarystyle: fixed bug	\tabularnewline 234
caused by using \ifdef in-	longheaderborder: switched to
stead of \ifcsdef 171	\tabularnewline 234
\gls@assign@desc@field:	\SetFootnoteAcronymDisplayStyle:
changed to use \glssetnoexpandfield	fixed missing argument bug 201
..... 15	super: switched to \tabularnewline
\gls@assign@descplural@field: 249
changed to use \glssetnoexpandfieldsuper3col:	switched to
..... 15	\tabularnewline 250
\gls@assign@name@field:	super3colheader: switched to
changed to use \glssetnoexpandfield	\tabularnewline 251
..... 15	super4col: switched to
\gls@assign@type@field:	\tabularnewline 252
changed to use \glssetexpandfield	super4colheader: switched to
..... 15	\tabularnewline 252
\gls@checkseeallowed: new .. 52	super4colheaderborder:
\glsaddallunused: set default to	switched to \tabularnewline
\@glo@types 141 253
\Glsentryfull: changed to use	superheader: switched to
\acrfullformat 138	\tabularnewline 249
\glsentryfull: changed to use	superheaderborder: switched to
\acrfullformat 138	\tabularnewline 250
\Glsentryfullpl: changed to 3.14a	
use \acrfullformat 139	\@glswritefiles: renamed
\glsentryfullpl: changed to	\glswritefiles to \@glswritefiles
use \acrfullformat 138	and used "savewrites" option
\glsglossarymark: renamed	to set \glswritefiles 151
\glossarymark to \glsglossarymark	General: new 216
to avoid conflict with memoir 31	acronyms: new 12
\glsprestandardsort: new 8	\gls@defglossaryentry: added
\glssetnoexpandfield: new .. 15	check for existence of default
altsuper4colheader: switched	glossary 63
to \tabularnewline 254	set the default for firstplural to
altsuper4colheaderborder:	be the value of plural 65
switched to \tabularnewline	xindygloss: new 20
..... 254	\longprovideglossaryentry:
long: switched to \tabularnewline	new 62
..... 233	compatible-2.07: added check
long3col: switched to \tabularnewline	for 2.07 before setting 3.07
..... 235	compatibility 21
long3colheader: switched to	notranslate: new 17
\tabularnewline 235	\provideglossaryentry: new . 56
long3colheaderborder: switched 4.0	
to \tabularnewline 236	\gls@defglossaryentry: added
long4col: switched to \tabularnewline	check for first key 65
..... 236 4.01	
long4colheader: switched to	General: fixed non-value options
\tabularnewline 237	so that they can be passed to

document class	5	xindynoglsnumbers:new	20
\CustomAcronymFields: in-		sm-short-long:new	184
serted missing comma	209	sm-short-long-desc:new ...	186
4.02		\makeglossaries: made pream-	
\@acrfull:now using \acrfullfmt		ble only	150
.....	176	index:new	22
\@gls@indexdef:new	22	\newacronymstyle:new	182
\@gls@numbersdef:new	22	long-sc-short:new	184
\@gls@symbolsdef:new	21	long-sc-short-desc:new ...	185
General: Removed \acronymfont		long-short:new	182
.....	130-133	long-short-desc:new	184
\ACRfullfmt:new	178	long-sm-short:new	184
\Acrfullfmt:new	177	long-sm-short-desc:new ...	185
\acrfullfmt:new	176	footnote:new	188
\ACRfullplfmt:new	179	footnote-desc:new	190
\Acrfullplfmt:new	179	footnote-sc:new	190
\acrfullplfmt:new	178	footnote-sc-desc:new	190
\acronymentry:new	181	footnote-sm:new	190
sanitize: fixed bug that caused		footnote-sm-desc:new	191
an error here	17	\setacronymstyle:new	181
sc-short-long:new	184	\SetDescriptionAcronymDisplayStyle:	
sc-short-long-desc:new ...	186	Moved check for empty cus-	
\Genacrfullformat:new	80	tom text to prevent unwanted	
\genacrfullformat:new	80	parenthetical material	199
\GenericAcronymFields:new	181	\SetDescriptionFootnoteAcronymDisplayStyle:	
\Genplacrfullformat:new ...	81	Moved check for empty cus-	
\genplacrfullformat:new ...	81	tom text to prevent unwanted	
\Glsentryfull: bug fix: added		parenthetical material	195
missing \acronymfont	138	\SetFootnoteAcronymDisplayStyle:	
\glentryfull: bug fix: added		Moved check for empty cus-	
missing \acronymfont	138	tom text to prevent unwanted	
\Glsentryfullpl: bug fix: added		parenthetical material	201
missing \acronymfont	138	\SetGenericNewAcronym:new	180
\glentryfullpl: bug fix: added		\SetSmallAcronymDisplayStyle:	
missing \acronymfont	138	Moved check for empty cus-	
\glsgenacfmt:new	78	tom text to prevent unwanted	
\GlsUseAcrEntryDispStyle:		parenthetical material	204
new	182	dua:new	186
\GlsUseAcrStyleDefs:new ..	182	dua-desc:new	188
short-long:new	183	numberedsection: added	
short-long-desc:new	185	nameref option	4

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	153
\@glossarysec	4
\@glossaryseclabel	4
\@glossarysecstar	4
\@gls@default@entryfmt	297
\@gls@expand@field	55
\@ACRlong	303
\@ACRshort	302
\@Acrlong	302
\@Acrshort	301
\@GLS@	98
\@GLSpl	102
\@Gls@	97
\@Glspl@	100
\@PGLS	221
\@PGLS@	221
\@PGLSpl	222
\@PGLSpl@	222
\@PglS	220
\@PglS@	220
\@PglSpl	220
\@PglSpl@	221
\@acrfull	176
\@acrlong	302
\@acrshort	301
\@addtoacronymlists	12
\@delimN	173
\@delimR	172
\@disable@onlypremakeg	24
\@disable@premakecs	24
\@disabled@glSaddxdycounters	35
\@do@seeglossary	156
\@do@wrglossary	153, 269
\@glo@seeautonumberlist	6
\@glo@storeentry	68
\@glo@types	47
\@glossary	152
\@glossary@default@style	5
\@glossaryentryfield	67
\@glossarysection	32
\@glossarysubentryfield	67
\@gls	95
\@gls@	95
\@gls@link	84
\@gls@addpredefinedattributes	37
\@gls@assign@symbol@field ...	15
\@gls@assign@symbolplural@field	15
\@gls@checkactual	92
\@gls@checkbar	91
\@gls@checkescactual	89
\@gls@checkescbar	90
\@gls@checkesclevel	90
\@gls@checkescquote	89
\@gls@checklevel	91
\@gls@checkmkidxchars	87
\@gls@checkquote	88
\@gls@codepage	42
\@gls@counterwithin	8
\@gls@declareoption	5
\@gls@default@value	51
\@gls@do@acronymsdef	12
\@gls@escbsdq	87
\@gls@expand@fields	55
\@gls@fixbraces	156
\@gls@getcounter	49
\@gls@getcounterprefix	155
\@gls@getgrouptitle	169
\@gls@hypergroup	226
\@gls@ifinlist	35
\@gls@indexdef	22
\@gls@keymap	58, 216
\@gls@link	84
\@gls@loadlist	6
\@gls@loadlong	6
\@gls@loadsuper	6
\@gls@loadtree	6
\@gls@makefirstuc	224
\@gls@missingnumberlist	54
\@gls@noaccess	291
\@gls@noexpand@field	54
\@gls@noexpand@fields	54
\@gls@nohyperlist	14
\@gls@notranslatorhook	17

<code>\Acf</code>	192	<code>long-sm-short-desc</code>	185
<code>\acf</code>	192	<code>sc-short-long</code>	184
<code>\Acfp</code>	192	<code>sc-short-long-desc</code>	186
<code>\acfp</code>	192	<code>short-long</code>	183
<code>\Acl</code>	192	<code>short-long-desc</code>	185
<code>\acl</code>	192	<code>sm-short-long</code>	184
<code>\Aclp</code>	192	<code>sm-short-long-desc</code>	186
<code>\aclp</code>	192	<code>\acronymentry</code>	181
<code>\Acp</code>	193	<code>\acronymfont</code>	
<code>\acp</code>	192 78, 179, 197, 201, 203, 206	
<code>\acrfootnote</code>	194	<code>acronymlists (option)</code>	13
<code>\ACRfull</code>	177	<code>\acronymname</code>	25
<code>\Acrfull</code>	177	<code>acronyms (option)</code>	12
<code>\acrfull</code>	176, 181, 189	<code>\acronymsort</code>	181
<code>\ACRfullfmt</code>	178	<code>\acronymtype</code>	11, 175
<code>\Acrfullfmt</code>	177	<code>\acrpluralsuffix</code>	175
<code>\acrfullfmt</code>	176	<code>\ACRshort</code>	127
<code>\acrfullformat</code>	78, 177	<code>\Acrshort</code>	126
<code>\ACRfullpl</code>	179	<code>\acrshort</code>	125
<code>\Acrfullpl</code>	178	<code>\ACRshortpl</code>	129
<code>\acrfullpl</code>	178	<code>\Acrshortpl</code>	128
<code>\ACRfullplfmt</code>	179	<code>\acrshortpl</code>	127
<code>\Acrfullplfmt</code>	179	<code>\Acs</code>	191
<code>\acrfullplfmt</code>	178	<code>\acs</code>	191
<code>\acrlinkfootnote</code>	194	<code>\Acsp</code>	191
<code>\acrlinkfullformat</code>	176	<code>\acsp</code>	191
<code>\ACRlong</code>	131	<code>\addglossarytocaptions</code>	26
<code>\Acrlong</code>	130	<code>\addto</code>	26
<code>\acrlong</code>	129	<code>align (environment)</code>	69, 85
<code>\ACRlongpl</code>	133	<code>altlist (style)</code>	231
<code>\Acrlongpl</code>	132	<code>altlistgroup (style)</code>	231
<code>\acrlongpl</code>	131	<code>altlisthypergroup (style)</code>	231
<code>\acrnameformat</code>	179, 200	<code>altlong4col (style)</code>	237
<code>\acrno linkfootnote</code>	195	<code>altlong4colborder (style)</code>	238
<code>acronym (option)</code>	11	<code>altlong4colheader (style)</code>	238
<code>acronym styles:</code>		<code>altlong4colheaderborder (style)</code>	238
<code>dua</code>	186	<code>altlongragged4col (style)</code>	242
<code>dua-desc</code>	188	<code>altlongragged4colborder (style)</code>	243
<code>footnote</code>	188	<code>altlongragged4colheader (style)</code>	243
<code>footnote-desc</code>	190	<code>altlongragged4colheaderborder</code>	
<code>footnote-sc</code>	190	(style)	244
<code>footnote-sc-desc</code>	190	<code>\altnewglossary</code>	48
<code>footnote-sm</code>	190	<code>altsuper4col (style)</code>	253
<code>footnote-sm-desc</code>	191	<code>altsuper4colborder (style)</code>	254
<code>long-sc-short</code>	184	<code>altsuper4colheader (style)</code>	254
<code>long-sc-short-desc</code>	185	<code>altsuper4colheaderborder</code>	
<code>long-short</code>	182	(style)	254
<code>long-short-desc</code>	184	<code>altsuperragged4col (style)</code>	258
<code>long-sm-short</code>	184		

`altsuperragged4colborder`
 (style) 260
`altsuperragged4colheader`
 (style) 259
`altsuperragged4colheaderborder`
 (style) 260
`almtree` (style) 265
`almtreegroup` (style) 267
`almtreehypergroup` (style) 268
`amsgen` package 2, 82
`amsmath` package 69
`\andname` 26
`array` package 239, 255
`article` class 155

B

`babel` package 23, 25, 26, 41, 310

C

`\capitalisewords` 224
`\changes` 3
`\compatglossarystyle` 275
`compatible-2.07` (option) 21
`compatible-3.07` (option) 21
`\compatibleglossentry` 166
`\compatiblesubglossentry` ... 167
`counter` (key) 51
`counter` (option) 13
`\CustomAcronymFields` 209
`\CustomNewAcronymDef` 209

D

`\DeclareAcronymList` 12
`\DefaultNewAcronymDef` .. 193, 305
`\defentryfmt` 82
`\defglsgdisplay` 81
`\defglsgdisplayfirst` 82
`\defglsgentry` 48
`\defglsgentryfmt` 47, 50, 51, 73
`\DefineAcronymSynonyms` 191
`\delimN` 30, 172
`\delimR` 30, 172
`description` (environment) 229, 230
`description` (key) 50
`description` (option) 18
`descriptionaccess` (key) 290
`\DescriptionDUANewAcronymDef` 197
`\DescriptionFootnoteNewAcronymDef`
 195, 305
`\descriptionname` 25

`\DescriptionNewAcronymDef` ..
 199, 306
`descriptionplural` (key) 50
`descriptionpluralaccess` (key) 290
`doc` package 2, 3, 10
`dua` (acrstyle) 186
`dua` (option) 19
`dua-desc` (acrstyle) 188
`\DUANewAcronymDef` 206

E

`entrycounter` (option) 7
`entrycounterwithin` (option) 8
`\entryname` 25
`environments`:
 `align` 69, 85
 `description` 229, 230
 `longtable` 6, 210, 232–244
 `multicols` 244
 `supertabular` ... 6, 210, 248–260
 `theglossary` 3, 30, 31,
 165, 171, 246, 247, 262, 263, 265
 `theindex` 261
`equation` counter 85, 86
`etoolbox` package 2, 222

F

`file` types
 `.aux` 160
 `.glo` 68
 `.ist` 141, 142, 148
 `.toc` 34
 `.xdy` 28
 `glo` 215
`\findrootlanguage` 41
`first` (key) 50
`firstaccess` (key) 289
`\firstacronymfont` 78, 179
`firstplural` (key) 51
`firstpluralaccess` (key) 289
`footnote` (acrstyle) 188
`footnote` (option) 18
`footnote-desc` (acrstyle) 190
`footnote-sc` (acrstyle) 190
`footnote-sc-desc` (acrstyle) 190
`footnote-sm` (acrstyle) 190
`footnote-sm-desc` (acrstyle) 191
`\FootnoteNewAcronymDef` . 202, 307
`\forallglossaries` 43
`\forallglsgentries` 43

\forglseentries 43

G

garamondx package 175

\Genacrfullformat 80

\genacrfullformat 80

\GenericAcronymFields 181

\Genplacrfullformat 81

\genplacrfullformat 81

\glolinkprefix 85

glossareentry counter 164

glossaries package

... 42, 142, 210, 216, 229, 268, 288

glossaries-accsupp package ... 67, 288

\GlossariesWarning 14

\GlossariesWarningNoLine 14

\glossary 47, 148, 152, 170

glossary counters:

glossaryentry 163

glossarysubentry 163

glossary keys:

access 289

counter 51

description 50

descriptionaccess 290

descriptionplural 50

descriptionpluralaccess . 290

first 50

firstaccess 289

firstplural 51

firstpluralaccess 289

long 53

longaccess 290

longplural 53

longpluralaccess 290

name 50

nonumberlist 52

parent 52

plural 50

pluralaccess 289

see 52

short 53

shortaccess 290

shortplural 53

shortpluralaccess 290

sort 50

symbol 51

symbolaccess 290

symbolplural 51

symbolpluralaccess 290

text 50

textaccess 289

type 51

user1 52

user2 52

user3 52

user4 52

user5 53

user6 53

glossary package 1, 174

glossary styles:

altlist 231, 276

altlist 231

altlistgroup 231, 276

altlistgroup 231

altlisthypergroup ... 231, 276

altlisthypergroup 231

altlong4col .. 237, 238, 242, 278

altlong4col 237

altlong4colborder ... 238, 278

altlong4colborder 238

altlong4colheader ... 238, 278

altlong4colheader 238

altlong4colheaderborder .

..... 238, 279

altlong4colheaderborder . 238

altlongragged4col 242, 243, 280

altlongragged4col 242

altlongragged4colborder .

..... 243, 280

altlongragged4colborder . 243

altlongragged4colheader .

..... 243, 280

altlongragged4colheader . 243

altlongragged4colheaderborder

..... 244, 280

altlongragged4colheaderborder

..... 244

altsuper4col . 253, 254, 258, 287

altsuper4col 253

altsuper4colborder .. 254, 288

altsuper4colborder 254

altsuper4colheader .. 254, 288

altsuper4colheader 254

altsuper4colheaderborder

..... 254, 288

altsuper4colheaderborder 254

altsuperragged4col 258–260, 286

altsuperragged4col	258
altsuperragged4colborder	260, 286
altsuperragged4colborder	260
altsuperragged4colheader	259, 286
altsuperragged4colheader	259
altsuperragged4colheaderborder	260, 286
altsuperragged4colheaderborder	260
alttree	247, 265, 267, 282
alttree	265
alttreegroup	268, 283
alttreegroup	267
alttreehypergroup ...	268, 283
alttreehypergroup	268
index	244, 260–262, 280
index	260
indexgroup	261, 262, 281
indexgroup	261
indexhypergroup	262, 281
indexhypergroup	262
inline	275
inline	227
list	5, 229–232, 276
list	229
listdotted	232, 276
listdotted	232
listgroup	230, 276
listgroup	230
listhypergroup	230, 276
listhypergroup	230
long	233, 234, 239, 277, 279
long	233
long3col	234, 235, 277
long3col	234
long3colborder	235, 277
long3colborder	235
long3colheader	235, 278
long3colheader	235
long3colheaderborder	235, 278
long3colheaderborder	235
long4col	236, 237, 278
long4col	236
long4colborder	237, 278
long4colborder	237
long4colheader	236, 278
long4colheader	236
long4colheaderborder	237, 278
long4colheaderborder	237
longborder	233, 277
longborder	233
longheader	234, 277
longheader	234
longheaderborder	234, 277
longheaderborder	234
longragged	239–241
longragged	239
longragged3col ...	241, 242, 279
longragged3col	241
longragged3colborder	241, 279
longragged3colborder	241
longragged3colheader	242, 279
longragged3colheader	242
longragged3colheaderborder	242, 280
longragged3colheaderborder	242
longraggedborder	240, 279
longraggedborder	240
longraggedheader	240, 279
longraggedheader	240
longraggedheaderborder	240, 279
longraggedheaderborder ..	240
mcolalttree	247, 284
mcolalttree	247
mcolalttreegroup	247, 284
mcolalttreegroup	247
mcolalttreehypergroup	247, 284
mcolalttreehypergroup ...	247
mcolindex	245, 283
mcolindex	244
mcolindexgroup	245, 284
mcolindexgroup	245
mcolindexhypergroup .	245, 284
mcolindexhypergroup	245
mcoltree	245, 284
mcoltree	245
mcoltreegroup	284
mcoltreegroup	245
mcoltreehypergroup ..	246, 284
mcoltreehypergroup	246
mcoltreenoname	246, 284
mcoltreenoname	246
mcoltreenonamegroup .	246, 284
mcoltreenonamegroup	246

mcoltreenonamehypergroup	246, 284	tree	262
mcoltreenonamehypergroup	246	treegroup	246, 263, 281
sublistdotted	277	treegroup	263
sublistdotted	232	treehypergroup	263, 281
super	248–250, 256, 286	treehypergroup	263
super	248	treenoname	246, 264, 282
super3col	250, 251, 286	treenoname	264
super3col	250	treenonamegroup	265, 282
super3colborder	251, 287	treenonamegroup	264
super3colborder	251	treenonamehypergroup	265, 282
super3colheader	251, 287	treenonamehypergroup	265
super3colheader	251	glossary-hypernav package	142
super3colheaderborder	251, 287	glossary-list package	5, 6, 229
super3colheaderborder	251	glossary-long package	
super4col	252, 253, 287		6, 232, 233, 242, 248
super4col	252	glossary-longragged package	239
super4colborder	253, 287	glossary-mcols package	244
super4colborder	253	glossary-super package	
super4colheader	252, 287		6, 233, 248, 255, 258
super4colheader	252	glossary-superragged package	255
super4colheaderborder	253, 287	glossary-tree package	7, 260
super4colheaderborder	253	glossaryentry (counter)	163
superborder	249, 286	glossaryentry counter	7, 8, 164, 165
superborder	249	\glossaryentryfield	167, 171
superheader	249, 286	\glossaryentrynumber	162, 163
superheader	249	\glossaryentrynumbers	
superheaderborder	250, 286		5, 30, 158, 161
superheaderborder	250	\glossaryheader	165, 171
superragged	255–257, 285	\glossarymark	32
superragged	255	\glossaryname	25, 26
superragged3col	257, 258, 285	\glossarypostamble	31, 171
superragged3col	257	\glossarypreamble	30, 171
superragged3colborder	257, 285	\glossarysection	4, 31, 47
superragged3colborder	257	\glossarystyle	170, 210
superragged3colheader	258, 285	glossarysubentry (counter)	163
superragged3colheader	258	glossarysubentry counter	8, 163–165
superragged3colheaderborder	258, 285	\glossarysubentryfield	168
	256, 285	\glossentry	51, 166
superraggedborder	256, 285	\Glossentrydesc	167
superraggedborder	256	\glossentrydesc	166, 303
superraggedheader	256, 285	\Glossentryname	166
superraggedheader	256	\glossentryname	166, 303
superraggedheaderborder	256, 285	\Glossentrysymbol	167
	256, 285	\glossentrysymbol	167, 303
superraggedheaderborder	256	\GLS	98
superraggedright3colheaderborder	258	\Gls	96, 100, 223
tree	245, 262–265, 281	\gls	2,
			51, 71, 83, 95, 97, 99, 104, 105,

107-109, 111, 112, 114, 115, 117-119, 121, 122, 124, 164, 218	
\gls@Alphpage	153
\gls@alphpage	153
\gls@assign@desc	61
\gls@assign@desc@field	15
\gls@assign@descplural@field	15
\gls@assign@field	56
\gls@assign@name@field	15
\gls@assign@type@field	15
\gls@checkisacronymlist	13
\gls@checkseeallowed	52
\gls@codepage	20
\gls@defglossaryentry	62
\gls@disablepagerefexpansion	153
\gls@doclearpage	33
\gls@doentryfmt	47
\gls@hypergroup prerun	226
\gls@ifnotmeasuring	69
\gls@level	54
\gls@numberpage	153
\gls@protected@pagefmts	153
\gls@Romanpage	153
\gls@romanpage	153
\gls@save@numberlist	158
\gls@suffiX	29
\gls@suffiXFF	29
\glsaccessdisplay	297
\glsaccsupp	294
\glsadd	71, 140, 170
\glsadd options	
counter	140
format	140, 172
\glsaddall	71, 141
\glsaddall options	
types	140, 141
\glsaddallunused	141
\glsaddkey	58
\GlsAddLetterGroup	43
\GlsAddSortRule	41
\GlsAddXdyAlphabet	37
\GlsAddXdyAttribute	36, 268
\GlsAddXdyCounters	35, 269
\GlsAddXdyLocation	39, 269
\GlsAddXdyStyle	41
\glsautoprefix	4
\glsclearpage	34
\glsclosebrace	142
\glscompositor	28, 39
\glscounter	13, 48
\GlsDeclareNoHyperList	14
\glsdefaulttype	11
\glsdefmain	10
\GLSdesc	112
\Glsdesc	111
\glsdesc	111, 112
\GLSdescplural	113
\Glsdescplural	113
\glsdescplural	112, 113
\glsdescriptionaccessdisplay	295
\glsdescriptionpluralaccessdisplay	296
\glsdescwidth ...	233, 239, 248, 255
\glsdisablehyper	94
\glsdisp	103
\glsdisplay	72, 81, 95
\glsdisplayfirst	72, 81, 95
\glsdisplaynumberlist	139
\glsdoifexists	44
\glsdoifnoexists	45
\glsdoparenifnotempty	203
\glsenablehyper	94
\glsentryaccess	292
\glsentrycounter	85
\glsentrycounterlabel	164
\Glsentrydesc	134
\glsentrydesc	134
\glsentrydescaccess	293
\Glsentrydescplural	135
\glsentrydescplural	134
\glsentrydescpluralaccess ..	293
\Glsentryfirst	136
\glsentryfirst	136
\glsentryfirstaccess	292
\Glsentryfirstplural	136
\glsentryfirstplural	136
\glsentryfirstpluralaccess ..	293
\glsentryfmt	50, 51, 72
\Glsentryfull	138
\glsentryfull	138, 181, 189
\Glsentryfullpl	139
\glsentryfullpl	138
\glsentryitem	165
\Glsentrylong	138
\glsentrylong	138
\glsentrylongaccess	293
\Glsentrylongpl	138
\glsentrylongpl	138

<code>\glentrylongpluralaccess</code> ..	294	<code>\GLSfirstplural</code>	109
<code>\Glsentryname</code>	134	<code>\Glsfirstplural</code>	109
<code>\glentryname</code>	134, 157	<code>\glsfirstplural</code>	108, 109
<code>\glentrynumberlist</code>	139	<code>\glsfirstpluralaccessdisplay</code>	295
<code>\Glsentryplural</code>	135	<code>\glsgenacfmt</code>	78
<code>\glentryplural</code>	135	<code>\glsgenentryfmt</code>	76
<code>\glentrypluralaccess</code>	293	<code>\glsgetgrouplabel</code>	170
<code>\Glsentryprefix</code>	217	<code>\glsgetgrouptitle</code>	142, 169
<code>\glentryprefix</code>	217	<code>\glsglossarymark</code>	7, 31
<code>\Glsentryprefixfirst</code>	217	<code>\glsgroupheading</code>	168, 171
<code>\glentryprefixfirst</code>	217	<code>\glsgroupskip</code>	168, 171, 229
<code>\Glsentryprefixfirstplural</code> .	217	<code>\glshyperlink</code>	140
<code>\glentryprefixfirstplural</code> .	217	<code>\glshypernavsep</code>	227
<code>\Glsentryprefixplural</code>	218	<code>\glshypernumber</code>	30, 172
<code>\glentryprefixplural</code>	217	<code>\glsIfListOfAcronyms</code>	12
<code>\Glsentryshort</code>	138	<code>\glsinlinedescformat</code>	229
<code>\glentryshort</code>	137	<code>\glsinlinedopostchild</code> ..	227, 228
<code>\glentryshortaccess</code>	293	<code>\glsinlineemptydescformat</code> ..	229
<code>\Glsentryshortpl</code>	138	<code>\glsinlinenameformat</code>	229
<code>\glentryshortpl</code>	138	<code>\glsinlineparentchildseparator</code>	
<code>\glentryshortpluralaccess</code> .	293	229
<code>\glentrysort</code>	136	<code>\glsinlinepostchild</code>	229
<code>\Glsentrysymbol</code>	135	<code>\glsinlineseparator</code>	229
<code>\glentrysymbol</code>	135	<code>\glsinlinesubdescformat</code>	229
<code>\glentrysymbolaccess</code>	293	<code>\glsinlinesubnameformat</code>	229
<code>\Glsentrysymbolplural</code>	135	<code>\glsinlinesubseparator</code>	229
<code>\glentrysymbolplural</code>	135	<code>\glskeylisttok</code>	180
<code>\glentrysymbolpluralaccess</code>	293	<code>\glslabeltok</code>	180
<code>\Glsentrytext</code>	135	<code>\glslink</code> .	71, 72, 83, 95, 140, 170, 172
<code>\glentrytext</code>	135, 157	<code>\glslink options</code>	
<code>\glentrytextaccess</code>	292	counter	82, 95, 216
<code>\glentrytype</code>	136	format	83, 95, 172
<code>\Glsentryuseri</code>	136	hyper	83, 95
<code>\glentryuseri</code>	136	local	83
<code>\Glsentryuserii</code>	137	<code>\glslistdottedwidth</code>	232
<code>\glentryuserii</code>	136	<code>\glslocalreset</code>	70
<code>\Glsentryuseriii</code>	137	<code>\glslocalresetall</code>	71
<code>\glentryuseriii</code>	137	<code>\glslocalunset</code>	70
<code>\Glsentryuseriv</code>	137	<code>\glslocalunsetall</code>	71
<code>\glentryuseriv</code>	137	<code>\glslongaccessdisplay</code>	296
<code>\Glsentryuserv</code>	137	<code>\glslongaccesskey</code>	308
<code>\glentryuserv</code>	137	<code>\glslongkey</code>	176
<code>\Glsentryuservi</code>	137	<code>\glslongpluralaccessdisplay</code>	296
<code>\glentryuservi</code>	137	<code>\glslongpluralaccesskey</code>	308
<code>\glsexpandfields</code>	56	<code>\glslongpluralkey</code>	176
<code>\GLSfirst</code>	106	<code>\glslongtok</code>	180
<code>\Glsfirst</code>	106	<code>\glsmakefirstuc</code>	224
<code>\glsfirst</code>	105, 106	<code>\glsmcols</code>	244
<code>\glsfirstaccessdisplay</code>	295	<code>\glsmoveentry</code>	67

<code>\GLSname</code>	110	<code>\glsSetCompositor</code>	28, 29
<code>\Glsname</code>	110	<code>\glssetnoexpandfield</code>	15
<code>\glsname</code>	109, 110	<code>\glsSetSuffixF</code>	29
<code>\glsnameaccessdisplay</code>	294	<code>\glsSetSuffixFF</code>	29
<code>\glsnamefont</code>	171	<code>\glssettoctitle</code>	25
<code>\glsnavhyperlink</code>	225	<code>\glssetwidest</code>	265
<code>\glsnavhypertarget</code>	225	<code>\GlsSetXdyCodePage</code>	42
<code>\glsnavigation</code>	226	<code>\GlsSetXdyFirstLetterAfterDigits</code>	142
<code>\glsnextpages</code>	163	<code>\GlsSetXdyLanguage</code>	42
<code>\glsnoexpandfields</code>	56	<code>\GlsSetXdyLocationClassOrder</code>	40
<code>\glsnonextpages</code>	163	<code>\GlsSetXdyMinRangeLength</code>	142
<code>\glsnoxindywarning</code>	34	<code>\GlsSetXdyStyles</code>	41
<code>\glsnumberformat</code>	30	<code>\glsshortaccessdisplay</code>	296
<code>\glsnumbersgroupname</code>	25, 142, 169	<code>\glsshortaccesskey</code>	308
<code>\glsnumlistlastsep</code>	140	<code>\glsshortkey</code>	176
<code>\glsnumlistsep</code>	140	<code>\glsshortpluralaccessdisplay</code>	296
<code>\glsopenbrace</code>	142	<code>\glsshortpluralaccesskey</code>	308
<code>\glsorder</code>	19	<code>\glsshortpluralkey</code>	176
<code>\glsorg@endtheglossary</code>	3	<code>\glsshorttok</code>	180
<code>\glsorg@glossary</code>	2	<code>\glssortnumberfmt</code>	9
<code>\glsorg@theglossary</code>	3	<code>\glsstepentry</code>	164
<code>\glsorg@wrglossary</code>	3	<code>\glsstepsubentry</code>	164
<code>\glspagelistwidth</code>	233, 239, 248, 255	<code>\glssubentrycounterlabel</code>	165
<code>\glspar</code>	28	<code>\glssubentryitem</code>	165
<code>\GLSpl</code>	101	<code>\GLSsymbol</code>	115
<code>\Glspl</code>	100, 223	<code>\Glsymbol</code>	114
<code>\glspl</code>	71, 99–101	<code>\glsymbol</code>	114, 115
<code>\GLSplural</code>	108	<code>\glsymbolaccessdisplay</code>	295
<code>\Glsplural</code>	107	<code>\glsymbolnav</code>	227
<code>\glsplural</code>	107, 108	<code>\GLSsymbolplural</code>	116
<code>\glspluralaccessdisplay</code>	294	<code>\Glsymbolplural</code>	116
<code>\glspluralsuffix</code>	25, 50, 51	<code>\glsymbolplural</code>	115, 116
<code>\glspostdescription</code>	7	<code>\glsymbolpluralaccessdisplay</code>	295
<code>\glspostinline</code>	229	<code>\glsymbolsgroupname</code>	25, 142, 169
<code>\glsprestandardsort</code>	8	<code>\glstarget</code>	165
<code>\glsquote</code>	142	<code>\GLStext</code>	104
<code>\glsrefentry</code>	164	<code>\Glstext</code>	105
<code>\glsreset</code>	69	<code>\glstext</code>	104
<code>\glsresetall</code>	70	<code>\glstextaccessdisplay</code>	294
<code>\glsresetentrylist</code>	163	<code>\glstextformat</code>	72
<code>\glsresetsubentrycounter</code>	163, 164	<code>\glstextup</code>	175
<code>\glssee</code>	156	<code>\glstreeindent</code>	263, 264
<code>\glsseeformat</code>	144, 156	<code>\glsunset</code>	70
<code>\glsseeitem</code>	157	<code>\glsunsetall</code>	71
<code>\glsseeitemformat</code>	157	<code>\GlsUseAcrEntryDispStyle</code>	182
<code>\glsseelastsep</code>	157	<code>\GlsUseAcrStyleDefs</code>	182
<code>\glsseelist</code>	157	<code>\GLSuseri</code>	118
<code>\glsseesep</code>	157		
<code>\glsSetAlphaCompositor</code>	29		

<code>\Glsuseri</code>	117
<code>\glsuseri</code>	117, 118
<code>\GLSuserii</code>	119
<code>\Glsuserii</code>	119
<code>\glsuserii</code>	118, 119
<code>\GLSuseriii</code>	120
<code>\Glsuseriii</code>	120
<code>\glsuseriii</code>	120
<code>\GLSuseriv</code>	122
<code>\Glsuseriv</code>	121
<code>\glsuseriv</code>	121, 122
<code>\GLSuserv</code>	123
<code>\Glsuserv</code>	123
<code>\glsuserv</code>	122, 123
<code>\GLSuservi</code>	125
<code>\Glsuservi</code>	124
<code>\glsuservi</code>	124, 125
<code>\glswrite</code>	151
<code>\glswritedefhook</code>	61

H

<code>\hyperbf</code>	174
<code>\hyperemph</code>	174
<code>hyperfirst</code> (option)	18
<code>\hyperit</code>	174
<code>\hyperlink</code>	94
<code>\hypermd</code>	174
<code>\hyperpage</code>	172
<code>hyperref</code> package ...	155, 158, 172, 216
<code>\hyperrm</code>	173
<code>\hypersc</code>	174
<code>\hypersf</code>	174
<code>\hypersl</code>	174
<code>\hypertarget</code>	94
<code>\hypertt</code>	174
<code>\hyperup</code>	174

I

<code>\if@gls@docloaded</code>	2
<code>\if@gls@isacronymlist</code>	13
<code>\ifglossaryexists</code>	44
<code>\ifglsdescsuppressed</code>	45
<code>\ifglsentryexists</code>	44
<code>\ifglshaschildren</code>	45
<code>\ifglshasdesc</code>	45
<code>\ifglshaslong</code>	46
<code>\ifglshasparent</code>	45
<code>\ifglshasprefix</code>	218
<code>\ifglshasprefixfirst</code>	218
<code>\ifglshasprefixfirstplural</code> ..	218

<code>\ifglshasprefixplural</code>	218
<code>\ifglshasshort</code>	46
<code>\ifglshassymbol</code>	46
<code>\ifglstranslate</code>	17
<code>\ifglsused</code>	44, 69
<code>\ifglsxindy</code>	19
<code>index</code> (option)	22
<code>index</code> (style)	260
<code>indexgroup</code> (style)	261
<code>indexhypergroup</code> (style)	262
<code>indexonlyfirst</code> (option)	18
<code>inline</code> (style)	227
<code>\inputencodingname</code>	20
<code>\istfile</code>	151
<code>\istfilename</code>	28
<code>\item</code>	171, 229, 261

L

<code>link text</code>	72
<code>list</code> (style)	229
<code>listdotted</code> (style)	232
<code>listgroup</code> (style)	230
<code>listhypergroup</code> (style)	230
<code>\loadglsentries</code>	11, 72
<code>long</code> (key)	53
<code>long</code> (style)	233
<code>long-sc-short</code> (acrstyle)	184
<code>long-sc-short-desc</code> (acrstyle) ..	185
<code>long-short</code> (acrstyle)	182
<code>long-short-desc</code> (acrstyle)	184
<code>long-sm-short</code> (acrstyle)	184
<code>long-sm-short-desc</code> (acrstyle) ..	185
<code>long3col</code> (style)	234
<code>long3colborder</code> (style)	235
<code>long3colheader</code> (style)	235
<code>long3colheaderborder</code> (style) ..	235
<code>long4col</code> (style)	236
<code>long4colborder</code> (style)	237
<code>long4colheader</code> (style)	236
<code>long4colheaderborder</code> (style) ..	237
<code>longaccess</code> (key)	290
<code>longborder</code> (style)	233
<code>longheader</code> (style)	234
<code>longheaderborder</code> (style)	234
<code>\longnewglossaryentry</code>	50, 61
<code>longplural</code> (key)	53
<code>longpluralaccess</code> (key)	290
<code>\longprovideglossaryentry</code> ...	62
<code>longragged</code> (style)	239

longragged3col (style) 241
 longragged3colborder (style) .. 241
 longragged3colheader (style) .. 242
 longragged3colheaderborder
 (style) 242
 longraggedborder (style) 240
 longraggedheader (style) 240
 longraggedheaderborder (style) 240
 longtable (environment)
 6, 210, 232–244
 longtable package 232, 239

M

\makefirstuc 222
 makeglossaries .. 19, 28, 42, 47, 160
 \makeglossaries
 24, 28, 29, 46, 48, 149, 150
 \makeglossary 150
 makeindex
 .. 8, 19, 25, 28–30, 47, 49, 50,
 69, 86, 89, 141, 142, 144, 147,
 148, 154, 155, 168, 169, 269, 270
 delim_n 30
 delim_r 30
 page_compositor 28
 special characters 87, 88, 141
 makeindex (option) 19
 mcolalttree (style) 247
 mcolalttreegroup (style) 247
 mcolalttreehypergroup (style) . 247
 mcolindex (style) 244
 mcolindexgroup (style) 245
 mcolindexhypergroup (style) ... 245
 mcoltree (style) 245
 mcoltreegroup (style) 245
 mcoltreehypergroup (style) 246
 mcoltreenoname (style) 246
 mcoltreenonamegroup (style) ... 246
 mcoltreenonamehypergroup
 (style) 246
 memoir class 152
 mfirstuc package 1
 \mfirstucMakeUppercase 224
 multicol package 244
 multicol (environment) 244

N

name (key) 50
 \new@glossaryentry 56
 \newacronym 18, 19, 53, 71, 72, 174, 175

\newacronymhook 180
 \newacronymstyle 182
 \newglossary 13, 47, 49, 148, 150, 161
 \newglossaryentry 50, 56, 71, 72, 175
 \newglossaryentry options
 access 291, 292
 counter 51
 description 18, 49, 50,
 53, 56, 62, 111, 134, 175, 202, 290
 descriptionaccess 293, 295
 descriptionplural 112, 290
 descriptionpluralaccess .. 293, 296
 first 50, 51,
 65, 95, 105, 136, 200, 205, 206, 289
 firstaccess 292, 295
 firstplural 51, 108, 136, 289
 firstpluralaccess 293, 295
 format 143
 long 78, 138, 290
 longaccess 293, 296
 longplural 138, 290
 longpluralaccess 294, 296
 name 49,
 50, 53, 56, 62, 109, 134, 157, 289
 nonumberlist 52
 parent 52, 56
 plural 50, 65, 107, 289
 pluralaccess 293, 294
 prefix 217
 prefixfirst 217
 prefixfirstplural 217
 prefixplural 217
 see 6, 52, 150
 short 78, 137, 290
 shortaccess 293, 296
 shortplural 138, 290
 shortpluralaccess 293, 296
 sort 50, 136, 168, 169
 symbol ... 49, 51, 114, 135, 196,
 198, 200, 205, 236, 252, 289–291
 symbolaccess 293, 295
 symbolplural 115, 290
 symbolpluralaccess 293, 295
 text . 50, 95, 104, 135, 196, 200, 289
 textaccess 292, 294
 type 11, 51, 71, 136
 user1 117, 136, 290
 user2 118, 136
 user3 119, 137

user4	121, 137
user5	122, 137
user6	124, 137, 290
\newglossarystyle	171
nogroupskip (option)	7
nohypertypes (option)	14
\noist	148, 215, 274
nolist (option)	6
nolong (option)	6
nomain (option)	11
nonumberlist (key)	52
nonumberlist (option)	5
\nopostdesc	27
nopostdot (option)	7
nostyles (option)	7
nosuper (option)	6
notranslate (option)	17
notree (option)	7
nowarn (option)	14
numberedsection (option)	4
numberline (option)	4
numbers (option)	21

O

\oldacronym	174
order (option)	19

P

package options:	
acronym	11, 25, 161, 175
true	12
acronym	11
acronymlists	13
acronyms	12
compatible-2.07	21
compatible-3.07	21
counter	13
counter	13
description	200, 201
description	18
dua	199–201
dua	19
entrycounter	163
true	8
entrycounter	7
entrycounterwithin	8
footnote	96–
99, 101–103, 196, 199, 200, 202	
footnote	18

hyperfirst	
false	96–99, 101–103
hyperfirst	18
index	22
indexonlyfirst	18
makeindex	145, 216
makeindex	19
nogroupskip	7
nohypertypes	14
nolist	210
nolist	6
nolong	210, 233
nolong	6
nomain	10, 11
nomain	11
nonumberlist	5
nonumberlist	5
nopostdot	7
nostyles	7
nosuper	210
nosuper	6
notranslate	17
notree	210
notree	7
nowarn	14
numberedsection	4
numberline	4
numberline	4
numbers	21
order	19
sanitize	16, 49, 134, 135
sanitize	17
sanitizesort	16
savenumberlist	6
savewrites	20
false	149
true	151
savewrites	20
section	4, 32
section	4
seeautonumberlist	6
shotcuts	19
smallcaps	19
smaller	19
sort	
def	8, 9
standard	8
use	8, 9
sort	8

style	5, 210		
style	5		
subentrycounter	163		
subentrycounter	8		
symbols	21		
toc	4		
true	4		
toc	4		
translate	17		
false	17		
translate	17		
translator	17		
ucmark	7		
xindy	20, 145, 216		
xindy	20		
xindygloss	20		
xindynoglsnumbers	20		
\pagelistname	25		
parent (key)	52		
\PGLS	221		
\Pgls	219		
\pgls	218		
\PGLSpl	222		
\Pglspl	220		
\pglspl	219		
\phantomsection	31–33		
plural (key)	50		
pluralaccess (key)	289		
polyglossia package	23, 26		
\printacronyms	11		
\PrintChanges	3		
\printglossaries			
.	10, 30, 46, 49, 150, 158, 161, 225		
\printglossary			
... ..	30, 31, 47, 150, 158, 161, 225		
\printglossary options			
nogroupskip	162		
nonumberlist	162		
numberedsection	162		
style	161		
title	161		
toctitle	161		
type	11, 157, 161		
\provideglossaryentry	56		
R			
\renewglossarystyle	171		
\roman	38		
S			
sanitize (option)	17		
sanitizesort (option)	16		
savenumberlist (option)	6		
savewrites (option)	20		
sc-short-long (acrstyle)	184		
sc-short-long-desc (acrstyle) ..	186		
section (option)	4		
see (key)	52		
seeautonumberlist (option)	6		
\seename	26		
\SetAcronymLists	13		
\SetAcronymStyle	12, 208		
\setacronymstyle	181		
\SetCustomDisplayStyle	209		
\SetCustomStyle	209		
\SetDefaultAcronymDisplayStyle			
.....	193		
\SetDefaultAcronymStyle	194		
\SetDescriptionAcronymDisplayStyle			
.....	199		
\SetDescriptionAcronymStyle	200		
\SetDescriptionDUAAcronymDisplayStyle			
.....	197		
\SetDescriptionDUAAcronymStyle			
.....	198		
\SetDescriptionFootnoteAcronymDisplayStyle			
.....	195		
\SetDescriptionFootnoteAcronymStyle			
.....	196		
\SetDUADisplayStyle	206		
\SetDUASyle	207		
\setentrycounter	170		
\SetFootnoteAcronymDisplayStyle			
.....	201		
\SetFootnoteAcronymStyle ...	202		
\SetGenericNewAcronym	180		
\setglossarypreamble	30		
\setglossarysection	32		
\setglossarystyle	170		
\setglossentrycompatibility	167		
\SetSmallAcronymDisplayStyle	204		
\SetSmallAcronymStyle	205		
\setStyleFile	28		
\setupglossaries	22		
short (key)	53		
short-long (acrstyle)	183		
short-long-desc (acrstyle)	185		
shortaccess (key)	290		

`\theHequation` 155
`theindex (environment)` 261
`toc (option)` 4
`\translate` 26
`translate (option)` 17
`translator package`
 23, 26, 158, 310, 319–322
`tree (style)` 262
`treegroup (style)` 263
`treehypergroup (style)` 263
`treenoname (style)` 264
`treenonamegroup (style)` 264
`treenonamehypergroup (style)` .. 265
`type (key)` 51

U

`ucmark (option)` 7
`user1 (key)` 52
`user2 (key)` 52
`user3 (key)` 52

`user4 (key)` 52
`user5 (key)` 53
`user6 (key)` 53

W

`\warn@nomakeglossaries` 149
`\warn@noprintglossary` 158
`\writeist` 28, 35, 37, 40, 143, 268, 270

X

`\xcapitalisewords` 224
`\xglsacccsupp` 294
`xindy` 8, 19, 20, 28, 29, 34, 37,
 39, 41–43, 68, 92, 93, 142–144,
 154, 155, 160, 168, 215, 269, 270
`xindy (option)` 20
`xindygloss (option)` 20
`xindynoglsnumbers (option)` 20
`\xmakefirstuc` 224
`xspace package` 2, 174