

Documented Code For glossaries v4.06

Nicola L.C. Talbot

Dickimaw Books

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

2014-03-12

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

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

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

[glossaries-user.pdf](#) For the main user guide, read “glossaries.sty v4.06: \TeX 2e Package to Assist Generating Glossaries”.

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

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

INSTALL Installation instructions.

CHANGES Change log.

README Package summary.

Contents

1 Main Package Code	3
1.1 Package Definition	3
1.2 Package Options	5
1.3 Default values	29
1.4 Xindy	38
1.5 Loops and conditionals	47
1.6 Defining new glossaries	53
1.7 Defining new entries	55
1.8 Resetting and unsetting entry flags	76
1.9 Loading files containing glossary entries	78
1.10 Using glossary entries in the text	78
1.10.1 Links to glossary entries	89
1.10.2 Displaying entry details without adding information to the glossary	138
1.11 Adding an entry to the glossary without generating text	145
1.12 Creating associated files	146
1.13 Writing information to associated files	160
1.14 Glossary Entry Cross-References	165
1.15 Displaying the glossary	167
1.16 Acronyms	195
1.17 Predefined acronym styles	200
1.18 Predefined Glossary Styles	231
1.19 Debugging Commands	231
1.20 Compatibility with version 2.07 and below	237
2 Prefix Support (glossaries-prefix Code)	237
3 Mfirstuc Documented Code	244
4 Glossary Styles	246
4.1 Glossary hyper-navigation definitions (glossary-hypernav package)	246
4.2 In-line Style (glossary-inline.sty)	248
4.3 List Style (glossary-list.sty)	251
4.4 Glossary Styles using longtable (the glossary-long package)	254
4.5 Glossary Styles using longtable (the glossary-longragged package)	260
4.6 Glossary Styles using multicol (glossary-mcols.sty)	265
4.7 Glossary Styles using supertabular environment (glossary-super package)	269
4.8 Glossary Styles using supertabular environment (glossary-superragged package)	276
4.9 Tree Styles (glossary-tree.sty)	282
5 glossaries-compatible-207	289

6 Accessibility Support (glossaries-accsupp Code)	309
6.1 Defining Replacement Text	310
6.2 Accessing Replacement Text	314
6.3 Displaying the Glossary	329
6.4 Acronyms	330
6.5 Debugging Commands	345
7 Multi-Lingual Support	346
7.1 Babel Captions	346
7.2 Polyglossia Captions	352
7.3 Brazilian Dictionary	355
7.4 Danish Dictionary	356
7.5 Dutch Dictionary	356
7.6 English Dictionary	356
7.7 French Dictionary	357
7.8 German Dictionary	357
7.9 Irish Dictionary	357
7.10 Italian Dictionary	357
7.11 Magyar Dictionary	358
7.12 Polish Dictionary	358
7.13 Serbian Dictionary	358
7.14 Spanish Dictionary	359
Glossary	359
Change History	359
Index	379

1 Main Package Code

1.1 Package Definition

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

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries}[2014/03/12 v4.06 (NLCT)]
```

Required packages:

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

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

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

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

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

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

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

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

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

Check if doc has been loaded.

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

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

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

```
14 {%
```

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

```
16 }%
```

```
17 {%
```

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

```
19 }
```

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

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

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

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

```
22   \@bsphack
```

```
23   \begingroup
```

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

```
25 }
```

```
\glsorg@wrglossary
```

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

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

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

```
29   \endgroup
```

```
30   \@esphack
```

```
31 }
```

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

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

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

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

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

```
37 }
```

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

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

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

`\glsorg@theglossary`

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

`\glsorg@endtheglossary`

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

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

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

End of doc stuff.

```
55 \fi
```

1.2 Package Options

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

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

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

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

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

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

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

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

Determine whether or not to use numbered sections.

`\@@glossarysecstar`

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

`\@@glossaryseclabel`

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

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

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

`numberedsection`

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

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

`\@glossary@default@style`

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

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

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

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

`\@gls@declareoption`

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

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

`\glossaryentrynumbers`

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

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

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

savenumberlist Provide means to store the number list for entries.

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

`\@glo@seeautonumberlist`

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

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

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

```

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

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

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

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

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

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

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

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

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

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

```



```

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

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

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

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

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

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

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

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

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

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

```

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

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

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

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

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

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

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

Store entry information when it's defined.

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

No count register required for standard sort.

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

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

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

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

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

```
161     \fi
```

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

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

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

```
165   }%
```

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

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

```
167 }
```

Set standard sort as the default:

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

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

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

```

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

```

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

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

Store entry information when it's defined.

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

Defined count register associated with the glossary.

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

```
180 \expandafter\global
```

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

```
182 }%
```

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

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

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

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

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

```
187 \expandafter\glssortnumberfmt
```

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

```
189 }%
```

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

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

```
191 }
```

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

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

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

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

Defined count register associated with the glossary.

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

```
195 \expandafter\global
```

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

```
197 }%
```

Initialise the sort key to empty.

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

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

```
200 }%
```

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

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

Get the parent, if one exists

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

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

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

```
204 \else
```

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

```
206 \fi
```

Set index information for this entry

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

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

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

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

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

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

```
213 \expandafter\glssortnumberfmt
```

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

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

```
216 \fi
```

```
217 }%
```

```
218 }
```

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

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

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

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

```
222 \else
```

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

```
224 \fi
```

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

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

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

```
227 }%
```

```
228 }
```

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

`\glsdefaultttype`

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

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

`\acronymtype`

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

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

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

```
232   \let\glsdefaultttype\relax
```

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

```
234 }
```

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

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

```
236   \ifglsacronym
```

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

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

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

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

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

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

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

```
243       }%
```

```
244     }%
```

```
245   \else
```

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

```
247   \fi
```

```
248 }
```

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

```
249 \AtBeginDocument{%
```

```
250   \ifglsacronym
```

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

```
252       }{%
```

```
253     }%
```

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

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

```
256     }%
```

```
257   \fi
```

```
258 }
```

`@gls@do@acronymsdef` Set default value

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

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

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

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

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

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

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

\@addtoacronymlists

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

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

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

\glsIfListOfAcronyms

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

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

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

Internal command requires label and list to be expanded:

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

Not found

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

Found

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

`\if@glssisacronymlist` Convenient boolean.

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

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

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

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

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

`acronymlists`

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

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

`\glscounter`

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

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

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

```

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

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

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

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

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

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

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

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

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

```



```

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

```

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

\@gls@sanitizedesc

```

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

```

\glssetexpandfield

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

Sets field to always expand.

```

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

```

\glssetnoexpandfield

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

Sets field to never expand.

```

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

```

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

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

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

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

gn@descplural@field

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

\@gls@sanitizename

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

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

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

@gls@sanitizesymbol

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

assign@symbol@field

Don't expand symbol by default.

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

@symbolplural@field

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

Sanitizing stuff:

\@gls@sanitizesort

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

\@@gls@sanitizesort

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

@gls@nosanitizesort

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

@noidx@sanitizesort

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

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

@noidx@nosanitizesort

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

```

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

```

lsglsnoidxstripaccents

```

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

```

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

```

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

```

```

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

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

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

```

sanitizesort

```

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

```

Default setting:

```

475 \glssanitizesorttrue
476 \glssetnoexpandfield{sortvalue}%

```

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

```

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

```

```

479 \glsssetexpandfield{sortvalue}%
480 }

481 \define@choicekey[glS]{sanitize}{sort}{true,false}[true]{%
482 \setbool{glsssanitizesort}{#1}%
483 \ifglsssanitizesort
484 \glsssetnoexpandfield{sortvalue}%
485 \else
486 \glsssetexpandfield{sortvalue}%
487 \fi
488 \GlossariesWarning{sanitize={sort} package option
489 deprecated. Use sanitizesort instead}%
490 }

```

sanitize

```

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

```

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

```
501 \newif\ifglstranslate
```

`ls@nottranslatorhook`

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

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

```

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

```

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

```

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

```

```

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

```

Set the default value:

```

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

```

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

```

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

```

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

```

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

```

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

```

534 \newcommand*\@gls@setacrstyle{}

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

shortcuts Define acronym shortcuts.

```

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

```

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

```

560 \newcommand*{\glsorder}{word}

```

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

```

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

```

order

```

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

```

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

```

564 \newif\ifglxindy

```

The default is `makeindex`:

```

565 \glxindyfalse

```

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

```

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

```

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

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

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

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

Define key to set the language

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

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

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

Define a key to set the code page.

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

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

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

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

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

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

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

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

```
587 \define@boolkey{glossaries.sty}[gls]{savewrites}[true]{%
588   \ifglssavewrites
589     \renewcommand*{\glswritefiles}{\@glswritefiles}%
```



```

590 \else
591   \let\glswritefiles\@empty
592 \fi
593 }

```

Set default:

```

594 \glssavewritesfalse
595 \let\glswritefiles\@empty

```

compatible-3.07

```

596 \define@boolkey{glossaries.sty}[gls]{compatible-3.07}[true]{}
597 \boolfalse{glscpatible-3.07}

```

compatible-2.07

```

598 \define@boolkey{glossaries.sty}[gls]{compatible-2.07}[true]{%
  Also set 3.07 compatibility if this option is set.
599   \ifbool{glscpatible-2.07}%
600   {%
601     \booltrue{glscpatible-3.07}%
602   }%
603   }%
604 }
605 \boolfalse{glscpatible-2.07}

```

symbols Create a “symbols” glossary type

```

606 \@gls@declareoption{symbols}{%
607   \let\@gls@do@symbolsdef\@gls@symbolsdef
608 }

```

Default is not to define the symbols glossary:

```

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

```

\@gls@symbolsdef

```

610 \newcommand*{\@gls@symbolsdef}{%
611   \newglossary[slg]{symbols}{sls}{slo}{\glssymbolsgroupname}%
612   \newcommand*{\printsymbols}[1][\printglossary[type=symbols,##1]]%
  Define hook to set the toc title when translator is in use.
613   \newcommand*{\@gls@tr@set@symbols@toctitle}{%
614     \translatelet{\glossarytoctitle}{Symbols (glossaries)}%
615   }%
616 }%

```

numbers Create a “symbols” glossary type

```

617 \@gls@declareoption{numbers}{%
618   \let\@gls@do@numbersdef\@gls@numbersdef
619 }

```

Default is not to define the numbers glossary:

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

\@gls@numbersdef

```
621 \newcommand*{\@gls@numbersdef}{%
622   \newglossary[nlg]{numbers}{nls}{nlo}{\glsnumbersgroupname}%
623   \newcommand*{\printnumbers}[1][\printglossary[type=numbers,##1]]%
   Define hook to set the toc title when translator is in use.
624   \newcommand*{\gls@tr@set@numbers@toctitle}{%
625     \translatelet{\glossarytoctitle}{Numbers (glossaries)}%
626   }%
627 }%
```

index Create an “index” glossary type

```
628 \@gls@declareoption{index}{%
629   \let\@gls@do@indexdef\@gls@indexdef
630 }
```

Default is not to define index glossary:

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

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

```
632 \newcommand*{\@gls@indexdef}{%
633   \newglossary[ilg]{index}{ind}{idx}{\indexname}%
634   \newcommand*{\printindex}[1][\printglossary[type=index,##1]]%
635   \newcommand*{\newterm}[2][\%
636     \newglossaryentry{##2}%
637     {type={index},name={##2},description={\nopostdesc},##1}}
638 }%
```

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

```
639 \@for\CurrentOption :=\@declaredoptions\do{%
640   \ifx\CurrentOption\@empty
641   \else
642     \@expandtwoargs
643     \in@ {,\CurrentOption ,}{,\@classoptionslist,\@curroptions,}%
644     \ifin@
645       \@use@option
646       \expandafter \let\csname ds@\CurrentOption\endcsname\@empty
647     \fi
648   \fi
649 }
```

Now process options passed to the package:

```
650 \ProcessOptionsX
```

Load backward compatibility stuff:

```
651 \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:

```
652 \disable@keys{glossaries.sty}{compatible-2.07,%
653 xindy,xindygloss,xindynoglsnumbers,makeindex,%
654 acronym,translate,nottranslate,nolong,nosuper,notree,nostyles,nomain}
```

Now define `\setupglossaries`:

```
655 \newcommand*{\setupglossaries}[1]{%
656   \renewcommand*{\@gls@setacrstyle}{}%
657   \ifglsacrshortcuts
658     \def\@gls@setupshortcuts{\glsacrshortcutstrue}%
659   \else
660     \def\@gls@setupshortcuts{%
661       \ifglsacrshortcuts
662         \DefineAcronymSynonyms
663       \fi
664     }%
665   \fi
666   \glsacrshortcutsfalse
667   \let\@gls@do@numbersdef\relax
668   \let\@gls@do@symbolssdef\relax
669   \let\@gls@do@indexdef\relax
670   \let\@gls@do@acronymsdef\relax
671   \setkeys{glossaries.sty}{#1}%
672   \@gls@setacrstyle
673   \@gls@setupshortcuts
674   \@gls@do@acronymsdef
675   \@gls@do@numbersdef
676   \@gls@do@symbolssdef
677   \@gls@do@indexdef
678 }
```

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

```
679 \ifglstranslate
680   \@ifpackageloaded{polyglossia}%
681   {%
```

polyglossia fakes babel so need to check for polyglossia first.

```
682   }%
683   {%
684     \@ifpackageloaded{babel}%
685     {%
686       \IfFileExists{translator.sty}%
687       {%
688         \RequirePackage{translator}%
689       }%
690     }%
691   }%
692   {}
```

```

693 }
694 \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}`).

```

695 \ifthenelse{\equal{\glscounter}{section}}{%
696 {%
697   \ifcsundef{chapter}{}%
698   {%
699     \let\@gls@old@chapter\@chapter
700     \def\@chapter[#1]#2{\@gls@old@chapter[#1]{#2}}%
701     \ifcsundef{hyperdef}{}{\hyperdef{section}{\thesection}{}}}%
702   }%
703 }%
704 {}

```

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

```

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

```

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

```

706 \newcommand*{\@onlypremakeg}[1]{%
707   \ifx\@gls@onlypremakeg\@empty
708     \def\@gls@onlypremakeg{#1}%
709   \else
710     \expandafter\toks@\expandafter{\@gls@onlypremakeg}%
711     \edef\@gls@onlypremakeg{\the\toks@,\noexpand#1}%
712   \fi
713 }

```

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

```

714 \newcommand*{\@disable@onlypremakeg}{%
715   \@for\@thiscs:=\@gls@onlypremakeg\do{%
716     \expandafter\@disable@premakecs\@thiscs%
717 }}

```

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

```

718 \newcommand*{\@disable@premakecs}[1]{%
719   \def#1{\PackageError{glossaries}{\string#1\space may only be
720     used before \string\makeglossaries}{You can't use

```

```

721 \string#1\space after \string\makeglossaries}}}%
722 }

```

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

```
723 \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

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

\glssettoctitle Sets the TOC title for the given glossary.

```

725 \newcommand*\glssettoctitle[1]{%
726 \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

```
727 \providecommand*\entryname{Notation}
```

\descriptionname

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

\symbolname

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

\pagelistname

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

Labels for makeindex's symbol and number groups:

glssymbolsgroupname

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

glsnumbersgroupname

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

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

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

`\seename`

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

`\andname`

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

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

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

```
736 \newcommand*{\addglossarytocaptions}[1]{%
737   \ifcsundef{captions#1}{}%
738   {%
739     \expandafter\let\expandafter\@gls@tmp\csname captions#1\endcsname
740     \expandafter\toks@\expandafter{\@gls@tmp
741       \renewcommand*{\glossaryname}{\translate{Glossary}}}%
742     }%
743     \expandafter\edef\csname captions#1\endcsname{\the\toks@}%
744   }%
745 }

746 \ifglstranslate
```

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

```
747 \@ifpackageloaded{translator}{%
748   \usedictionary{glossaries-dictionary}%
749   \addglossarytocaptions{portuges}%
750   \addglossarytocaptions{portuguese}%
751   \addglossarytocaptions{brazil}%
752   \addglossarytocaptions{brazilian}%
753   \addglossarytocaptions{danish}%
754   \addglossarytocaptions{dutch}%
755   \addglossarytocaptions{afrikaans}%
756   \addglossarytocaptions{english}%
757   \addglossarytocaptions{UKenglish}%
758   \addglossarytocaptions{USenglish}%
759   \addglossarytocaptions{american}%
760   \addglossarytocaptions{australian}%
761   \addglossarytocaptions{british}%
762   \addglossarytocaptions{canadian}%
763   \addglossarytocaptions{newzealand}%
764   \addglossarytocaptions{french}%
765   \addglossarytocaptions{frenchb}%

```

```

766 \addglossarytocaptions{français}%
767 \addglossarytocaptions{acadian}%
768 \addglossarytocaptions{canadien}%
769 \addglossarytocaptions{german}%
770 \addglossarytocaptions{germanb}%
771 \addglossarytocaptions{austrian}%
772 \addglossarytocaptions{naustrian}%
773 \addglossarytocaptions{ngerman}%
774 \addglossarytocaptions{irish}%
775 \addglossarytocaptions{italian}%
776 \addglossarytocaptions{magyar}%
777 \addglossarytocaptions{hungarian}%
778 \addglossarytocaptions{polish}%
779 \addglossarytocaptions{spanish}%
780 \renewcommand*{\glssettoctitle}[1]{%
781   \ifcsdef{gls@tr@set@#1@toctitle}%
782     {%
783       \csuse{gls@tr@set@#1@toctitle}%
784     }%
785     {%
786       \def\glossarytoctitle{\csname @glotype@#1@title\endcsname}%
787     }%
788 }%
789 \renewcommand*{\glossaryname}{\translate{Glossary}}%
790 \renewcommand*{\acronymname}{\translate{Acronyms}}%
791 \renewcommand*{\entryname}{\translate{Notation (glossaries)}}%
792 \renewcommand*{\descriptionname}{%
793   \translate{Description (glossaries)}}%
794 \renewcommand*{\symbolname}{\translate{Symbol (glossaries)}}%
795 \renewcommand*{\pagelistname}{%
796   \translate{Page List (glossaries)}}%
797 \renewcommand*{\glssymbolsgroupname}{%
798   \translate{Symbols (glossaries)}}%
799 \renewcommand*{\glsnumbersgroupname}{%
800   \translate{Numbers (glossaries)}}%
801 }{%
802   \@ifpackageloaded{polyglossia}%
803   {\RequirePackage{glossaries-polyglossia}}%
804   {%
805     \@ifpackageloaded{babel}{%
806       \RequirePackage{glossaries-babel}}{}%
807   }}
808 \else
809   \@gls@notranslatorhook
810 \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.

```
811 \DeclareRobustCommand*\nopostdesc{}\}
```

`\nopostdesc` Suppress next description terminator.

```
812 \newcommand*\nopostdesc{%
813   \let\org@glspostdescription\glspostdescription
814   \def\glspostdescription{%
815     \let\glspostdescription\org@glspostdescription}%
816 }
```

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

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

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

```
818 \newcommand{\glspar}{\par}
```

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

```
819 \ifglsxindy
820   \newcommand{\setStyleFile}[1]{%
821     \renewcommand{\istfilename}{#1.xdy}}
822 \else
823   \newcommand{\setStyleFile}[1]{%
824     \renewcommand{\istfilename}{#1.ist}}
825 \fi
```

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

```
826 \@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`

```
827 \ifglsxindy
828   \def\istfilename{\jobname.xdy}
829 \else
830   \def\istfilename{\jobname.ist}
831 \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`

```
832 \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`

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

`\glsSetCompositor` Sets the compositor.

```
834 \newcommand*\glsSetCompositor[1]{%
835   \renewcommand*\glscompositor{#1}}
Only use before \makeglossaries
836 \@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 `<letter><compositor><number>`. 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.

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

`glsSetAlphaCompositor` Sets the alpha compositor.

```
838 \ifglsxindy
839   \newcommand*\glsSetAlphaCompositor[1]{%
840     \renewcommand*\@glsAlphacompositor{#1}}
841 \else
842   \newcommand*\glsSetAlphaCompositor[1]{%
843     \glsnoxindywarning\glsSetAlphaCompositor}
844 \fi
Can only be used before \makeglossaries
845 \@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.

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

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

```
847 \newcommand*\glsSetSuffixF[1]{%
848   \renewcommand*\gls@suffixF{#1}}
Only has an effect when used before \makeglossaries
849 \@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.

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

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

```

851 \newcommand*\glsSetSuffixFF}[1]{%
852   \renewcommand*\gls@suffixFF{#1}%
853 }
```

`\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”.

```

854 \ifcsundef{hyperlink}%
855 {%
856   \newcommand*\glsnumberformat}[1]{#1}%
857 }%
858 {%
859   \newcommand*\glsnumberformat}[1]{\glshypernumber{#1}}%
860 }
```

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`

```

861 \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`

```

862 \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`

```

863 \newcommand*\glossarypreamble{%
864   \csuse{@glossarypreamble@\currentglossary}%
865 }
```

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

Code provided by Michael Pock.

```
866 \newcommand{\setglossarypreamble}[2][\glsdefaulttype]{%
867   \ifglossaryexists{#1}{%
868     \csgdef{@glossarypreamble@#1}{#2}%
869   }{%
870     \GlossariesWarning{%
871       Glossary ‘#1’ is not defined%
872     }%
873   }%
874 }
```

The glossary postamble is given by `\glossarypostamble`. This is provided to allow the user to add something after the end of the `\glossary` 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`

```
875 \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`.

```
876 \newcommand*{\glossarysection}[2][\@gls@title]{%
877   \def\@gls@title{#2}%
878   \ifcsundef{phantomsection}%
879   {%
880     \@glossarysection{#1}{#2}%
881   }%
882   {%
883     \p@glossarysection{#1}{#2}%
884   }%

885   \glsglossarymark{\glossarytoctitle}%
886 }
```

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

```
887 \ifcsundef{glossarymark}%
888 {%
889   \newcommand{\glsglossarymark}[1]{\glossarymark{#1}}
```

```

890 }%
891 {%
892   \@ifclassloaded{memoir}
893   {%
894     \newcommand{\glsglossarymark}[1]{%
895       \ifglsucmark
896         \markboth{\memUHead{#1}}{\memUHead{#1}}%
897       \else
898         \markboth{#1}{#1}%
899       \fi
900     }
901   }%
902   {%
903     \newcommand{\glsglossarymark}[1]{%
904       \ifglsucmark
905         \@mkboth{\mfirstucMakeUppercase{#1}}{\mfirstucMakeUppercase{#1}}%
906       \else
907         \@mkboth{#1}{#1}%
908       \fi
909     }
910   }
911 }

```

`\glossarymark` Provided for backward compatibility:

```

912 \providecommand{\glossarymark}[1]{%
913   \ifglsucmark
914     \@mkboth{\mfirstucMakeUppercase{#1}}{\mfirstucMakeUppercase{#1}}%
915   \else
916     \@mkboth{#1}{#1}%
917   \fi
918 }

```

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`

```

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

```

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

`\@glossarysection`

```

921 \newcommand*\@glossarysection[2]{%
922   \ifdefempty\@glossarysecstar
923   {%

```

```

924 \csname\@@glossarysec\endcsname{#2}%
925 }%
926 {%
927 \csname\@@glossarysec\endcsname*{#2}%
928 \@gls@toc{#1}{\@@glossarysec}%
929 }%

```

Do automatic labelling if required

```

930 \@@glossaryseclabel
931 }

```

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.

\@p@glossarysection

```

932 \newcommand*\@p@glossarysection}[2]{%
933 \glsclearpage
934 \phantomsection
935 \ifdefempty\@@glossarysecstar
936 {%
937 \csname\@@glossarysec\endcsname{#2}%
938 }%
939 {%
940 \@gls@toc{#1}{\@@glossarysec}%
941 \csname\@@glossarysec\endcsname*{#2}%
942 }%

```

Do automatic labelling if required

```

943 \@@glossaryseclabel
944 }

```

\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.

```

945 \newcommand*\gls@doclearpage{%
946 \ifthenelse{\equal{\@@glossarysec}{chapter}}{%
947 {%
948 \ifcsundef{cleardoublepage}%
949 {%
950 \clearpage
951 }%
952 {%
953 \ifcsdef{if@openright}%
954 {%
955 \if@openright
956 \cleardoublepage
957 \else
958 \clearpage

```

```

959         \fi
960     }%
961     {%
962         \cleardoublepage
963     }%
964 }%
965 }%
966 {}%
967 }

```

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

```

968 \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
969 \newcommand*{\@gls@toc}[2]{%
970     \ifglstoc
971         \ifglsnumberline
972             \addcontentsline{toc}{#2}{\numberline{#1}}%
973         \else
974             \addcontentsline{toc}{#2}{#1}%
975         \fi
976     \fi
977 }

```

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

```

978 \newcommand*{\glsnoxindywarning}[1]{%
979     \GlossariesWarning{Not in xindy mode --- ignoring \string#1}%
980 }

```

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

```

981 \ifglsxindy
982     \edef\@xdyattributes{\string"default\string"}%
983 \fi

```

```

\@xdyattributelist  Comma-separated list of attributes.
984 \ifglxsindy
985   \edef\@xdyattributelist{}\%
986 \fi

\@xdylocref  Define list of markup location references.
987 \ifglxsindy
988   \def\@xdylocref{}
989 \fi

\@gls@ifinlist
990 \newcommand*\@gls@ifinlist[4]{%
991   \def\@do@ifinlist##1,##2\end@do@ifinlist{%
992     \def\@gls@listsuffix{##2}%
993     \ifx\@gls@listsuffix\@empty
994       #4%
995     \else
996       #3%
997     \fi
998   }%
999   \@do@ifinlist,##2,##1,\end@do@ifinlist
1000 }

\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.
1001 \ifglxsindy
1002   \newcommand*\@xdycounters{}\@glscounter}
1003   \newcommand*\GlsAddXdyCounters[1]{%
1004     \@for\@gls@ctr:=#1\do{%
        Check if already in list before adding.
1005       \edef\@do@addcounter{%
1006         \noexpand\@gls@ifinlist{\@gls@ctr}\@xdycounters}\}%
1007       {%
1008         \noexpand\edef\noexpand\@xdycounters{\@xdycounters,%
1009           \noexpand\@gls@ctr}%
1010       }%
1011     }%
1012     \@do@addcounter
1013   }
1014 }

    Only has an effect before \writeist:
1015   \@onlypremakeg\GlsAddXdyCounters
1016 \else
1017   \newcommand*\GlsAddXdyCounters[1]{%
1018     \glsnnoxindywarning\GlsAddXdyAttribute
1019   }
1020 \fi

```

d@gl saddx dy counters Counters must all be identified before adding attributes.

```

1021 \newcommand*{\@disabled@gl saddx dy counters}{%
1022   \PackageError{glossaries}{\string\GlsAddXdyCounters\space
1023   can't be used after \string\GlsAddXdyAttribute}{Move all
1024   occurrences of \string\GlsAddXdyCounters\space before the first
1025   instance of \string\GlsAddXdyAttribute}%
1026 }

```

\GlsAddXdyAttribute Adds an attribute.

```

1027 \ifgl sxindy

```

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

```

1028 \newcommand*{\@gl saddx dy attribute[2]}{%

```

Add to xindy attribute list

```

1029   \edef\@xdyattributes{\@xdyattributes ^^J \string"#1\string" ^^J
1030   \string"#2#1\string"}%

```

Add to xindy markup location.

```

1031   \expandafter\toks@\expandafter{\@xdylocref}%
1032   \edef\@xdylocref{\the\toks@ ^^J%
1033   (markup-locref
1034   :open \string"\string~n%
1035   \expandafter\string\csname gl sX#2X#1\endcsname
1036   \string" ^^J
1037   :close \string"\string" ^^J
1038   :attr \string"#2#1\string")}%

```

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

```

1039   \expandafter\gdef\csname gl sX#2X#1\endcsname##1##2{%
1040   \setentrycounter{##1}{#2}\csname #1\endcsname{##2}%
1041   }%
1042 }

```

High-level command:

```

1043 \newcommand*\GlsAddXdyAttribute[1]{%

```

Add to comma-separated attribute list

```

1044   \ifx\@xdyattributelist\@empty
1045   \edef\@xdyattributelist{#1}%
1046   \else
1047   \edef\@xdyattributelist{\@xdyattributelist,#1}%
1048   \fi

```

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

```

1049   \@for\@this@counter:=\@xdycounters\do{%
1050   \protected@edef\gl s@do@addx dy attribute{%
1051   \noexpand\@gl saddx dy attribute{#1}{\@this@counter}%
1052   }
1053   \gl s@do@addx dy attribute
1054   }%

```


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

```
1055 \let\GlsAddXdyCounters\@disabled@glsaddxdycounters
1056 }
```

Only has an effect before `\writeist`:

```
1057 \@onlypremakeg\GlsAddXdyAttribute
1058 \else
1059 \newcommand*\GlsAddXdyAttribute[1]{%
1060 \glsnoxindywarning\GlsAddXdyAttribute}
1061 \fi
```

`redefinedattributes` Add known attributes for all defined counters

```
1062 \ifglxindy
1063 \newcommand*{\@gls@addpredefinedattributes}{%
1064 \GlsAddXdyAttribute{glsnumberformat}
1065 \GlsAddXdyAttribute{textrm}
1066 \GlsAddXdyAttribute{textsf}
1067 \GlsAddXdyAttribute{texttt}
1068 \GlsAddXdyAttribute{textbf}
1069 \GlsAddXdyAttribute{textmd}
1070 \GlsAddXdyAttribute{textit}
1071 \GlsAddXdyAttribute{textup}
1072 \GlsAddXdyAttribute{textsl}
1073 \GlsAddXdyAttribute{textsc}
1074 \GlsAddXdyAttribute{emph}
1075 \GlsAddXdyAttribute{glshypernumber}
1076 \GlsAddXdyAttribute{hyperrrm}
1077 \GlsAddXdyAttribute{hypersf}
1078 \GlsAddXdyAttribute{hypertt}
1079 \GlsAddXdyAttribute{hyperbf}
1080 \GlsAddXdyAttribute{hypermd}
1081 \GlsAddXdyAttribute{hyperit}
1082 \GlsAddXdyAttribute{hyperup}
1083 \GlsAddXdyAttribute{hypersl}
1084 \GlsAddXdyAttribute{hypersc}
1085 \GlsAddXdyAttribute{hyperemph}
1086 }
1087 \else
1088 \let\@gls@addpredefinedattributes\relax
1089 \fi
```

`\@xdyuseralphabets` List of additional alphabets

```
1090 \def\@xdyuseralphabets{}
```

`\GlsAddXdyAlphabet` `\GlsAddXdyAlphabet{<name>}{<definition>}` adds a new alphabet called `<name>`.

The definition must use xindy syntax.

```
1091 \ifglxindy
1092 \newcommand*{\GlsAddXdyAlphabet}[2]{%
1093 \edef\@xdyuseralphabets{%
```

```

1094 \x@xdyuseralphabets ^^J
1095 (define-alphabet "#1" (#2))}}
1096 \else
1097 \newcommand*{\GlsAddXdyAlphabet}[2]{%
1098 \glsnxindywarning\GlsAddXdyAlphabet}
1099 \fi

```

This code is only required for xindy:

```

1100 \ifglsxindy

```

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

```

1101 \newcommand*{\@gls@xdy@locationlist}{%
1102 roman-page-numbers,%
1103 Roman-page-numbers,%
1104 arabic-page-numbers,%
1105 alpha-page-numbers,%
1106 Alpha-page-numbers,%
1107 Appendix-page-numbers,%
1108 arabic-section-numbers%
1109 }

```

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.

```

1110 \protected@edef\@gls@roman{\@roman{0}\string"
1111 \string"roman-numbers-lowercase\string" :sep \string"}}%
1112 \@onelevel@sanitize\@gls@roman
1113 \edef\@tmp{\string" \string"roman-numbers-lowercase\string"
1114 :sep \string"}%
1115 \@onelevel@sanitize\@tmp
1116 \ifx\@tmp\@gls@roman
1117 \expandafter
1118 \edef\csname @gls@xdy@Lclass@roman-page-numbers\endcsname{%
1119 \string"roman-numbers-lowercase\string"%
1120 }%
1121 \else
1122 \expandafter
1123 \edef\csname @gls@xdy@Lclass@roman-page-numbers\endcsname{
1124 :sep \string"\@gls@roman\string"%
1125 }%
1126 \fi

```

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

```

1127 \expandafter\def\csname @gls@xdy@Lclass@Roman-page-numbers\endcsname{%
1128 \string"roman-numbers-uppercase\string"%
1129 }%

```

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

```

1130 \expandafter\def\csname @gls@xdy@Lclass@arabic-page-numbers\endcsname{%
1131 \string"arabic-numbers\string"%
1132 }%

```

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

```

1133 \expandafter\def\csname @gls@xdy@Lclass@alpha-page-numbers\endcsname{%
1134 \string"alpha\string"%
1135 }%

```

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

```

1136 \expandafter\def\csname @gls@xdy@Lclass@Alpha-page-numbers\endcsname{%
1137 \string"ALPHA\string"%
1138 }%

```

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

```

1139 \expandafter\def\csname @gls@xdy@Lclass@Appendix-page-numbers\endcsname{%
1140 \string"ALPHA\string"
1141 :sep \string"\@glsAlphacompositor\string"
1142 \string"arabic-numbers\string"%
1143 }

```

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

```

1144 \expandafter\def\csname @gls@xdy@Lclass@arabic-section-numbers\endcsname{%
1145 \string"arabic-numbers\string"
1146 :sep \string"\glscompositor\string"
1147 \string"arabic-numbers\string"%
1148 }%

```

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

```

1149 \def\@xdyuserlocationdefs{}

```

xdyuserlocationnames List of additional user location names

```

1150 \def\@xdyuserlocationnames{}

```

End of xindy-only block:

```

1151 \fi

```

\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.)

```

1152 \ifglsxindy
1153 \newcommand*\GlsAddXdyLocation[3][{}]{%
1154 \def\@gls@tmp{#1}%

```

```

1155 \ifx\@gls@tmp\@empty
1156 \edef\@xdyuserlocationdefs{%
1157 \@xdyuserlocationdefs ^^J%
1158 (define-location-class \string"#2\string"^^J\space\space
1159 \space(:sep \string"{}\glsoopenbrace\string" #3
1160 :sep \string"\glsclosebrace\string"))
1161 }%
1162 \else
1163 \edef\@xdyuserlocationdefs{%
1164 \@xdyuserlocationdefs ^^J%
1165 (define-location-class \string"#2\string"^^J\space\space
1166 \space(:sep "\glsoopenbrace"
1167 #1
1168 :sep "\glsclosebrace\glsoopenbrace" #3
1169 :sep "\glsclosebrace"))
1170 }%
1171 \fi
1172 \edef\@xdyuserlocationnames{%
1173 \@xdyuserlocationnames^^J\space\space\space
1174 \string"#1\string"}%
1175 }

```

Only has an effect before `\writeist`:

```

1176 \@onlypremakeg\GlsAddXdyLocation
1177 \else
1178 \newcommand*{\GlsAddXdyLocation}[2]{%
1179 \glsnnoxindywarning\GlsAddXdyLocation}
1180 \fi

```

`ylocationclassorder` Define location class order

```

1181 \ifglxindy
1182 \edef\@xdylocationclassorder{^^J\space\space\space
1183 \string"roman-page-numbers\string"^^J\space\space\space
1184 \string"arabic-page-numbers\string"^^J\space\space\space
1185 \string"arabic-section-numbers\string"^^J\space\space\space
1186 \string"alpha-page-numbers\string"^^J\space\space\space
1187 \string"Roman-page-numbers\string"^^J\space\space\space
1188 \string"Alpha-page-numbers\string"^^J\space\space\space
1189 \string"Appendix-page-numbers\string"
1190 \@xdyuserlocationnames^^J\space\space\space
1191 \string"see\string"
1192 }
1193 \fi

```

Change the location order.

`yLocationClassOrder`

```

1194 \ifglxindy
1195 \newcommand*\GlsSetXdyLocationClassOrder[1]{%
1196 \def\@xdylocationclassorder{#1}}

```

```

1197 \else
1198   \newcommand*\GlsSetXdyLocationClassOrder[1]{%
1199     \glsnnoxindywarning\GlsSetXdyLocationClassOrder}
1200 \fi

\@xdysortrules   Define sort rules

1201 \ifglsxindy
1202   \def\@xdysortrules{}
1203 \fi

\GlsAddSortRule  Add a sort rule

1204 \ifglsxindy
1205   \newcommand*\GlsAddSortRule[2]{%
1206     \expandafter\toks@\expandafter{\@xdysortrules}%
1207     \protected@edef\@xdysortrules{\the\toks@ ^^J
1208       (sort-rule \string"#1\string" \string"#2\string"))}%
1209   }
1210 \else
1211   \newcommand*\GlsAddSortRule[2]{%
1212     \glsnnoxindywarning\GlsAddSortRule}
1213 \fi

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

1214 \ifglsxindy
1215   \def\@xdyrequiredstyles{tex}
1216 \fi

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

1217 \ifglsxindy
1218   \newcommand*\GlsAddXdyStyle[1]{%
1219     \edef\@xdyrequiredstyles{\@xdyrequiredstyles,#1}}%
1220 \else
1221   \newcommand*\GlsAddXdyStyle[1]{%
1222     \glsnnoxindywarning\GlsAddXdyStyle}
1223 \fi

\GlsSetXdyStyles  Reset the list of required styles

1224 \ifglsxindy
1225   \newcommand*\GlsSetXdyStyles[1]{%
1226     \edef\@xdyrequiredstyles{#1}}
1227 \else
1228   \newcommand*\GlsSetXdyStyles[1]{%
1229     \glsnnoxindywarning\GlsSetXdyStyles}
1230 \fi

```

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

```
1231 \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.

```
1232 \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.

```
1233 \ifglsxindy
1234   \newcommand*\GlsSetXdyLanguage[2][\glsdefaulttype]{%
1235     \ifglossaryexists{#1}{%
1236       \expandafter\def\csname @xdy@#1@language\endcsname{#2}%
1237     }{%
1238       \PackageError{glossaries}{Can't set language type for
1239         glossary type '#1' --- no such glossary}{%
1240         You have specified a glossary type that doesn't exist}}
1241   \else
1242     \newcommand*\GlsSetXdyLanguage[2][]{%
1243       \glsnoindywarning\GlsSetXdyLanguage}
1244 \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.

```
1245 \def\@gls@codepage#1#2{}
```

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

```
1246 \ifglsxindy
1247   \newcommand*\GlsSetXdyCodePage[1]{%
1248     \renewcommand*\@gls@codepage{#1}%
1249   }

   Suggested by egreg:
1250   \AtBeginDocument{%
1251     \ifx\@gls@codepage\@empty
1252       \@ifpackageloaded{fontspec}{\def\@gls@codepage{utf8}}{}%
1253     \fi
1254   }
```

```

1255 \else
1256   \newcommand*{\GlsSetXdyCodePage}[1]{%
1257     \glsnoxindywarning\GlsSetXdyCodePage}
1258 \fi

```

`\@xdylettergroups` Store letter group definitions.

```

1259 \ifglxindy
1260   \ifglx@xindy@glslnumbers
1261     \def\@xdylettergroups{(define-letter-group
1262       \string"glslnumbers\string"^^J\space\space\space
1263       :prefixes (\string"0\string" \string"1\string"
1264       \string"2\string" \string"3\string" \string"4\string"
1265       \string"5\string" \string"6\string" \string"7\string"
1266       \string"8\string" \string"9\string")^^J\space\space\space
1267       :before \string"@glslfirstletter\string")}
1268   \else
1269     \def\@xdylettergroups{}
1270   \fi
1271 \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.

```

1272 \newcommand*\GlsAddLetterGroup[2]{%
1273   \expandafter\toks@\expandafter{\@xdylettergroups}%
1274   \protected@edef\@xdylettergroups{\the\toks@^^J%
1275     (define-letter-group \string"#1\string"^^J\space\space\space#2)}%
1276 }%

```

1.5 Loops and conditionals

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

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

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

```

1277 \newcommand*\forallglossaries[3][\@glo@types]{%
1278   \@for#2:=#1\do{\ifx#2\@empty\else#3\fi}%
1279 }

```

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

```
\forglentries[<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.

```
1280 \newcommand*\forglentries[3][\glsldefaulttype]{%
```

```

1281 \edef\@glo@list{\csname glolist@#1\endcsname}%
1282 \@for#2:=\@glo@list\do
1283 {%
1284 \ifdefempty{#2}{#3}%
1285 }%
1286 }

```

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

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

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

```

1287 \newcommand*\forallglsentries[3][\@glo@types]{%
1288 \expandafter\forallglossaries\expandafter[#1]{\@this@glo@}%
1289 {%
1290 \forallglsentries[\@this@glo@]{#2}{#3}%
1291 }%
1292 }

```

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

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

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

```

1293 \newcommand{\ifglossaryexists}[3]{%
1294 \ifcsundef{@glo@type@#1@out}{#3}{#2}%
1295 }

```

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

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

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

```
\glsdetoklabel
```

```
1296 \newcommand*\glsdetoklabel[1]{#1}
```

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


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

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

```
1297 \newcommand{\ifglentryexists}[3]{%
1298   \ifcsundef{glo@\glstetoklabel{#1}@name}{#3}{#2}%
1299 }
```

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

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

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

```
1300 \newcommand*{\ifglused}[3]{%
1301   \ifbool{glo@\glstetoklabel{#1}@flag}{#2}{#3}%
1302 }
```

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

`\glstoifexists`

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

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

```
1303 \newcommand{\glstoifexists}[2]{%
1304   \ifglentryexists{#1}{#2}{%
1305     \PackageError{glossaries}{Glossary entry ‘\glstetoklabel{#1}’
1306       has not been defined}{You need to define a glossary entry before you
1307       can use it.}}%
1308 }
```

`\glstoifnoexists`

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

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

```
1309 \newcommand{\glstoifnoexists}[2]{%
1310   \ifglentryexists{#1}{%
1311     \PackageError{glossaries}{Glossary entry ‘\glstetoklabel{#1}’ has already
1312       been defined}{}}{#2}%
1313 }
```

`\glstoifexistsorwarn`

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

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

```

1314 \newcommand{\glsdoifexistsorwarn}[2]{%
1315   \ifglsentryexists{#1}{#2}{%
1316     \GlossariesWarning{Glossary entry ‘\glsdetoklabel{#1}’
1317       has not been defined}%
1318   }%
1319 }

```

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

```

1320 \newcommand{\ifglshaschildren}[3]{%
1321   \glsdoifexists{#1}%
1322   {%
1323     \def\do@glshaschildren{#3}%
1324     \edef\@gls@thislabel{\glsdetoklabel{#1}}%
1325     \expandafter\forGlsentries\expandafter
1326       [\csname glo@\@gls@thislabel @type\endcsname]
1327     {\glo@label}%
1328     {%
1329       \letcs\glo@parent{glo@\glo@label @parent}%
1330       \ifdefequal\@gls@thislabel\glo@parent
1331       {%
1332         \def\do@glshaschildren{#2}%
1333         \@endfortrue
1334       }%
1335     }%
1336   }%
1337   \do@glshaschildren
1338 }%
1339 }

```

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

```

1340 \newcommand{\ifglshasparent}[3]{%
1341   \glsdoifexists{#1}%
1342   {%
1343     \ifcempty{glo@\glsdetoklabel{#1}@parent}{#3}{#2}%
1344   }%
1345 }

```

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

```

1346 \newcommand*\ifglshasdesc[3]{%
1347   \ifcempty{glo@\glsdetoklabel{#1}@desc}%
1348   {#3}%
1349   {#2}%
1350 }

```

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

```

1351 \newcommand*\ifglstdescsuppressed}[3]{%
1352   \ifcsequal{glo@glstdetoklabel{#1}@desc}{@no@post@desc}%
1353   {#2}%
1354   {#3}%
1355 }

```

```

\ifglshassymbol \ifglshassymbol{<label>}{<true part>}{<false part>}
1356 \newcommand*\ifglshassymbol}[3]{%
1357   \letcs{\@glo@symbol}{glo@glstdetoklabel{#1}@symbol}%
1358   \ifdefempty\@glo@symbol
1359   {#3}%
1360   {%
1361     \ifdefequal\@glo@symbol\@gls@default@value
1362     {#3}%
1363     {#2}%
1364   }%
1365 }

```

```

\ifglshaslong \ifglshaslong{<label>}{<true part>}{<false part>}
1366 \newcommand*\ifglshaslong}[3]{%
1367   \letcs{\@glo@long}{glo@glstdetoklabel{#1}@long}%
1368   \ifdefempty\@glo@long
1369   {#3}%
1370   {%
1371     \ifdefequal\@glo@long\@gls@default@value
1372     {#3}%
1373     {#2}%
1374   }%
1375 }

```

```

\ifglshasshort \ifglshasshort{<label>}{<true part>}{<false part>}
1376 \newcommand*\ifglshasshort}[3]{%
1377   \letcs{\@glo@short}{glo@glstdetoklabel{#1}@short}%
1378   \ifdefempty\@glo@short
1379   {#3}%
1380   {%
1381     \ifdefequal\@glo@short\@gls@default@value
1382     {#3}%
1383     {#2}%
1384   }%
1385 }

```

```

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

```

```

1386 \newcommand*\ifglshasfield}[4]{%
1387   \glsdoifexists{#2}%
1388   {%
1389     \letcs{\@glo@thisvalue}{glo@glstdetoklabel{#2}@#1}%

```

First check supplied field label is defined.

```
1390     \ifdef\@glo@thisvalue
1391     {%
```

Is defined, so now check if empty.

```
1392     \ifdefempty\@glo@thisvalue
1393     {%
```

Is empty, so doesn't have field set.

```
1394         #4%
1395     }%
1396     {%
```

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

```
1397         \ifdefequal\@glo@thisvalue\@gls@default@value{#4}{#3}%
1398     }%
1399 }%
1400 {%
```

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

```
1401     \@gls@fetchfield{\@gls@thisfield}{#1}%
```

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

```
1402     \ifdef\@gls@thisfield
1403     {%
```

Is defined, so now check if empty.

```
1404         \letcs{\@glo@thisvalue}{glo@glsdetoklabel{#2}\@gls@thisfield}%
1405         \ifdefempty\@glo@thisvalue
1406         {%
```

Is empty so field hasn't been set.

```
1407             #4%
1408         }%
1409     {%
```

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

```
1410         \ifdefequal\@glo@thisvalue\@gls@default@value{#4}{#3}%
1411     }%
1412 }%
1413 {%
```

Not defined.

```
1414         \GlossariesWarning{Unknown entry field '#1'}%
1415         #4%
1416     }%
1417 }%
1418 }%
1419 }
```

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`

```
1420 \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.

```
1421 \newcommand*\@gls@provide@newglossary{%
```

```
1422   \protected@write\@auxout{}\string\providecommand\string\@newglossary[4]{}%
```

Only need to do this once.

```
1423   \let\@gls@provide@newglossary\relax
```

```
1424 }
```

`\defglsentryfmt`

Allow different glossaries to have different display styles.

```
1425 \newcommand*\defglsentryfmt}[2][\glsdefaulttype]{%
```

```
1426   \csgdef{gls@#1@entryfmt}{#2}%
```

```
1427 }
```

`\gls@doentryfmt`

```
1428 \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`

```
1429 \newcommand*\newglossary}[5][glg]{%
```

```
1430   \ifglossaryexists{#2}%
```

```
1431   {%
```

```
1432     \PackageError{glossaries}{Glossary type ‘#2’ already exists}{%
```

```
1433     You can't define a new glossary called ‘#2’ because it already
```

```

1434   exists}%
1435 }%
1436 {%

    Check if default has been set
1437   \ifundef\glsdefaulttype
1438   {%
1439     \gdef\glsdefaulttype{#2}%
1440   }%

    Add this to the list of glossary types:
1441   \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.
1442   \expandafter\gdef\csname glolist@#2\endcsname{,}%

    Store details of this new glossary type:
1443   \expandafter\def\csname @glotype@#2@in\endcsname{#3}%
1444   \expandafter\def\csname @glotype@#2@out\endcsname{#4}%
1445   \expandafter\def\csname @glotype@#2@title\endcsname{#5}%

1446   \@gls@provide@newglossary
1447   \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.

1448   \ifcsundef{gls@#2@entryfmt}%
1449   {%
1450     \defglsentryfmt[#2]{\glsentryfmt}%
1451   }%
1452   {}%

    Define sort counter if required:
1453   \@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.)
1454   \@ifnextchar[{\@gls@setcounter{#2}}%
1455     {\@gls@setcounter{#2}[\glscounter]}}%
1456 }

\altnewglossary
1457 \newcommand*\altnewglossary}[3]{%
1458   \newglossary[#2-glg]{#1}{#2-gls}{#2-glo}{#3}%
1459 }

```

Only define new glossaries in the preamble:

```
1460 \@onlypreamble{\newglossary}
```

Only define new glossaries before `\makeglossaries`

```
1461 \@onlypremakeg\newglossary
```

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

```
\@newglossary
```

```
1462 \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
```

```
1463 \def\@gls@setcounter#1[#2]{%
```

```
1464   \expandafter\def\csname @gls@#1@counter\endcsname{#2}%
```

Add counter to xindy list, if not already added:

```
1465   \ifglsxindy
```

```
1466     \GlsAddXdyCounters{#2}%
```

```
1467   \fi
```

```
1468 }
```

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

```
\@gls@getcounter
```

```
1469 \newcommand*{\@gls@getcounter}[1]{%
```

```
1470   \csname @gls@#1@counter\endcsname
```

```
1471 }
```

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

```
1472 \glsdefmain
```

Define the “acronym” glossaries if required.

```
1473 \@gls@do@acronymsdef
```

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

```
1474 \@gls@do@symbolsdef
```

```
1475 \@gls@do@numbersdef
```

```
1476 \@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.

```
1477 \define@key{glossentry}{name}{%
1478 \def\@glo@name{#1}%
1479 }
```

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`.

```
1480 \define@key{glossentry}{description}{%
1481 \def\@glo@desc{#1}%
1482 }
```

descriptionplural

```
1483 \define@key{glossentry}{descriptionplural}{%
1484 \def\@glo@descplural{#1}%
1485 }
```

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

```
1486 \define@key{glossentry}{sort}{%
1487 \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.

```
1488 \define@key{glossentry}{text}{%
1489 \def\@glo@text{#1}%
1490 }
```

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.

```
1491 \define@key{glossentry}{plural}{%
1492 \def\@glo@plural{#1}%
1493 }
```


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.

```
1494 \define@key{glossentry}{first}{%
1495 \def\@glo@first{#1}%
1496 }
```

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.

```
1497 \define@key{glossentry}{firstplural}{%
1498 \def\@glo@firstplural{#1}%
1499 }
```

`\@gls@default@value`

```
1500 \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).

```
1501 \define@key{glossentry}{symbol}{%
1502 \def\@glo@symbol{#1}%
1503 }
```

symbolplural

```
1504 \define@key{glossentry}{symbolplural}{%
1505 \def\@glo@symbolplural{#1}%
1506 }
```

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

```
1507 \define@key{glossentry}{type}{%
1508 \def\@glo@type{#1}}
```

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

```
1509 \define@key{glossentry}{counter}{%
1510 \ifcsundef{c@#1}%
1511 {%
1512 \PackageError{glossaries}%
1513 {There is no counter called ‘#1’}%
1514 {%
1515 The counter key should have the name of a valid counter
1516 as its value%
```

```

1517 }%
1518 }%
1519 {%
1520 \def\@glo@counter{#1}%
1521 }%
1522 }

```

see The see key specifies a list of cross-references

```

1523 \define@key{glossentry}{see}{%
1524 \gls@checkseeallowed
1525 \def\@glo@see{#1}%
1526 \@glo@seeautonumberlist
1527 }

```

gls@checkseeallowed

```

1528 \newcommand*{\gls@checkseeallowed}{%
1529 \PackageError{glossaries}%
1530 {'see' key may only be used after \string\makeglossaries\space
1531 or \string\makenoidxglossaries}%
1532 {You must use \string\makeglossaries\space
1533 or \string\makenoidxglossaries\space before defining
1534 any entries that have a 'see' key}%
1535 }

```

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

```

1536 \define@key{glossentry}{parent}{%
1537 \def\@glo@parent{#1}}

```

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

```

1538 \define@choicekey{glossentry}{nonumberlist}[\val\nr]{true,false}[true]{%
1539 \ifcase\nr\relax
1540 \def\@glo@prefix{\glsnonextpages}%
1541 \else
1542 \def\@glo@prefix{\glsnextpages}%
1543 \fi
1544 }

```

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

user1

```

1545 \define@key{glossentry}{user1}{%
1546 \def\@glo@useri{#1}%
1547 }

```

user2

```

1548 \define@key{glossentry}{user2}{%
1549 \def\@glo@userii{#1}%
1550 }

```

user3

```
1551 \define@key{glossentry}{user3}{%
1552   \def\@glo@useriii{#1}%
1553 }
```

user4

```
1554 \define@key{glossentry}{user4}{%
1555   \def\@glo@useriv{#1}%
1556 }
```

user5

```
1557 \define@key{glossentry}{user5}{%
1558   \def\@glo@userv{#1}%
1559 }
```

user6

```
1560 \define@key{glossentry}{user6}{%
1561   \def\@glo@uservi{#1}%
1562 }
```

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.

```
1563 \define@key{glossentry}{short}{%
1564   \def\@glo@short{#1}%
1565 }
```

shortplural This key is provided for use by \newacronym.

```
1566 \define@key{glossentry}{shortplural}{%
1567   \def\@glo@shortpl{#1}%
1568 }
```

long This key is provided for use by \newacronym.

```
1569 \define@key{glossentry}{long}{%
1570   \def\@glo@long{#1}%
1571 }
```

longplural This key is provided for use by \newacronym.

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

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

```
1575 \newcommand*\@glsnname{%
1576   \PackageError{glossaries}{name key required in
1577     \string\newglossaryentry\space for entry '@@glo@label'}{You
1578     haven't specified the entry name}}
```

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

```
1579 \newcommand*\@glsnodedesc{%
1580   \PackageError{glossaries}
1581   {%
1582     description key required in \string\newglossaryentry\space
1583     for entry '\@glo@label'%
1584   }%
1585   {%
1586     You haven't specified the entry description%
1587   }%
1588 }%
```

\@glsdefaultplural Now obsolete. Don't use.

```
1589 \newcommand*\@glsdefaultplural{}
```

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

```
1590 \newcommand*\@gls@missingnumberlist}[1]{%
1591   ??%
1592   \ifglssavenumberlist
1593     \GlossariesWarning{Missing number list for entry '#1'.
1594       Maybe makeglossaries + rerun required.}%
1595   \else
1596     \PackageError{glossaries}%
1597     {Package option 'savenumberlist=true' required.}%
1598     {%
1599       You must use the 'savenumberlist' package option
1600       to reference location lists.%
1601     }%
1602   \fi
1603 }
```

\@glsdefaultsort Define command to set default sort.

```
1604 \newcommand*\@glsdefaultsort{\@glo@name}
```

\gls@level Register to increment entry levels.

```
1605 \newcount\gls@level
```

@gls@noexpand@field

```
1606 \newcommand{\@gls@noexpand@field}[3]{%
1607   \expandafter\global\expandafter
1608   \let\csname glo@#1@#2\endcsname#3%
1609 }
```

gls@noexpand@fields

```
1610 \newcommand{\@gls@noexpand@fields}[4]{%
1611   \ifcsdef{gls@assign@#3@field}
1612     {%
1613       \ifdefequal{#4}{\@gls@default@value}%
1614     }%
1615   }%
```

```

1614     {%
1615         \edef\@gls@value{\expandonce{#1}}%
1616         \csuse{gls@assign@#3@field}{#2}{\@gls@value}%
1617     }%
1618     {%
1619         \csuse{gls@assign@#3@field}{#2}{#4}%
1620     }%
1621 }%
1622 {%
1623     \ifdefequal{#4}{\@gls@default@value}%
1624     {%
1625         \edef\@gls@value{\expandonce{#1}}%
1626         \@gls@noexpand@field{#2}{#3}{\@gls@value}%
1627     }%
1628     {%
1629         \@gls@noexpand@field{#2}{#3}{#4}%
1630     }%
1631 }%
1632 }

```

\@gls@expand@field

```

1633 \newcommand{\@gls@expand@field}[3]{%
1634   \expandafter
1635   \protected@xdef\csname glo@#1@#2\endcsname{#3}%
1636 }

```

@gls@expand@fields

```

1637 \newcommand{\@gls@expand@fields}[4]{%
1638   \ifcsdef{gls@assign@#3@field}
1639   {%
1640       \ifdefequal{#4}{\@gls@default@value}%
1641       {%
1642           \edef\@gls@value{\expandonce{#1}}%
1643           \csuse{gls@assign@#3@field}{#2}{\@gls@value}%
1644       }%
1645       {%
1646           \expandafter\@gls@startswitexpandonce#4\relax\relax\gls@endcheck
1647           {%
1648               \@gls@expand@field{#2}{#3}{#4}%
1649           }%
1650           {%
1651               \csuse{gls@assign@#3@field}{#2}{#4}%
1652           }%
1653       }%
1654   }%
1655   {%
1656       \ifdefequal{#4}{\@gls@default@value}%
1657       {%

```

```

1658      \@gls@expand@field{#2}{#3}{#1}%
1659    }%
1660    {%
1661      \@gls@expand@field{#2}{#3}{#4}%
1662    }%
1663  }%
1664 }

```

startswithexpandonce

```

1665 \def\@gls@expandonce{\expandonce}
1666 \def\@gls@startswithexpandonce#1#2\gls@endcheck#3#4{%
1667   \def\@gls@tmp{#1}%
1668   \ifdefequal{\@gls@expandonce}{\@gls@tmp}{#3}{#4}%
1669 }

```

`\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.

```

1670 \let\gls@assign@field\@gls@expand@fields

```

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

```

1671 \newcommand*\glsexpandfields{%
1672   \let\gls@assign@field\@gls@expand@fields
1673 }

```

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

```

1674 \newcommand*\glsnoexpandfields{%
1675   \let\gls@assign@field\@gls@noexpand@fields
1676 }

```

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

```

1677 \newrobustcmd{\newglossaryentry}[2]{%

```

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

```

1678   \glsdoifnoexists{#1}%
1679   {%
1680     \gls@defglossaryentry{#1}{#2}%
1681   }%
1682 }

```

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

```

1683 \newrobustcmd{\provideglossaryentry}[2]{%
1684   \ifglentryexists{#1}%
1685   {%
1686   {%
1687     \gls@defglossaryentry{#1}{#2}%
1688   }%
1689 }
1690 \@onlypreamble{\provideglossaryentry}

```

`\newglossaryentry` For use in document environment.

```

1691 \newrobustcmd{\newglossaryentry}[2]{%
1692   \ifundef\@gls@deffile
1693   {%
1694     \global\newwrite\@gls@deffile
1695     \immediate\openout\@gls@deffile=\jobname.glsdefs
1696   }%
1697   {%
1698     \ifglentryexists{#1}{%
1699     {%
1700       \gls@defglossaryentry{#1}{#2}%
1701     }%
1702     \@gls@writedef{#1}%
1703   }
1704 \AtBeginDocument
1705 {
1706   \makeatletter
1707   \InputIfFileExists{\jobname.glsdefs}{%}{%
1708   \makeatother
1709   \let\newglossaryentry\newglossaryentry
1710 }
1711 \AtEndDocument{\ifdef\@gls@deffile{\closeout\@gls@deffile}{%}}

```

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

```

1712 \newcommand*{\@gls@writedef}[1]{%
1713   \immediate\write\@gls@deffile
1714   {%
1715     \string\ifglentryexists{#1}{%\expandafter\@gobble\string\%^~J%
1716     \expandafter\@gobble\string\{\expandafter\@gobble\string\%^~J%
1717     \string\gls@defglossaryentry{\glsdetoklabel{#1}}\expandafter
1718     \@gobble\string\%^~J%
1719     \expandafter\@gobble\string\{\expandafter\@gobble\string\%%
1720   }%

```

Write key value information:

```

1721 \@for\@gls@map:=\@gls@keymap\do
1722 {%
1723   \edef\glo@value{\expandafter\expandonce
1724     \csname glo@glsdetoklabel{#1}\@expandafter
1725     \@secondoftwo\@gls@map\endcsname}%
1726   \@onelevel@sanitize\glo@value

```

```

1727 \immediate\write\@gls@deffile
1728 {%
1729 \expandafter\@firstoftwo\@gls@map
1730 =\expandafter\@gobble\string\{\glo@value\expandafter\@gobble\string\},%
1731 \expandafter\@gobble\string\%%
1732 }%
1733 }%

```

Provide hook:

```

1734 \glswritedefhook
1735 \immediate\write\@gls@deffile
1736 {%
1737 \expandafter\@gobble\string\%^~J%
1738 \expandafter\@gobble\string\}\expandafter\@gobble\string\%^~J%
1739 \expandafter\@gobble\string\}\expandafter\@gobble\string\%%
1740 }%
1741 }

```

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

```

1742 \newcommand*{\@gls@keymap}{%
1743 {name}{name},%
1744 {sort}{sortvalue},% unescaped sort value
1745 {type}{type},%
1746 {first}{first},%
1747 {firstplural}{firstpl},%
1748 {text}{text},%
1749 {plural}{plural},%
1750 {description}{desc},%
1751 {descriptionplural}{descplural},%
1752 {symbol}{symbol},%
1753 {symbolplural}{symbolplural},%
1754 {user1}{useri},%
1755 {user2}{userii},%
1756 {user3}{useriii},%
1757 {user4}{useriv},%
1758 {user5}{userv},%
1759 {user6}{uservi},%
1760 {long}{long},%
1761 {longplural}{longpl},%
1762 {short}{short},%
1763 {shortplural}{shortpl},%
1764 {counter}{counter},%
1765 {parent}{parent}%
1766 }

```

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

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

```
1767 \newcommand*\@gls@fetchfield}[2]{%
```

Ensure user field name is fully expanded

```
1768 \edef\@gls@thisval{#2}%
```

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

```
1769 \@for\@gls@map:=\@gls@keymap\do{%
1770 \edef\@this@key{\expandafter\@firstoftwo\@gls@map}%
1771 \ifdefequal{\@this@key}{\@gls@thisval}%
1772 {%
```

Found it.

```
1773 \edef#1{\expandafter\@secondoftwo\@gls@map}%
```

Break out of loop.

```
1774 \@endfortrue
1775 }%
1776 {}%
1777 }%
1778 }
```

```
\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.

```
1779 \newcommand*\@glsaddkey{\@ifstar\@sglsaddkey\@glsaddkey}
```

Starred version switches on expansion for this key.

```
1780 \newcommand*\@sglsaddkey[#1]{%
1781 \key@ifundefined{glossentry}{#1}%
1782 {%
1783 \expandafter\newcommand\expandafter*\expandafter
1784 {\csname gls@assign@#1@field\endcsname}[2]{%
1785 \@gls@expand@field{##1}{#1}{##2}%
1786 }%
1787 }%
1788 {}%
1789 \@glsaddkey{#1}%
1790 }
```

Unstarred version doesn't override default expansion.

```
1791 \newcommand*\@glsaddkey[#1]{%
```

Check the specified key doesn't already exist.

```
1792 \key@ifundefined{glossentry}{#1}%
1793 {%
```

Set up the key.

```
1794 \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
1795 \appto\@gls@keymap{, #1}{#1}}%
```

Set the default value.

```
1796 \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
```

Assignment code.

```
1797 \appto\@newglossaryentryposthook{%
1798 \letcs{\@glo@tmp}{@glo@#1}%
1799 \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
1800 }%
```

Define the no-link commands.

```
1801 \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
1802 \newcommand*{#4}[1]{\@Gls@entry@field{##1}{#1}}%
```

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

```
1803 \ifcsdef{@gls@user@#1@}%
1804 {%
1805 \PackageError{glossaries}%
1806 {Can't define '\string#5' as helper command
1807 '\expandafter\string\csname @gls@user@#1@\endcsname' already exists}%
1808 }%
1809 }%
1810 {%
1811 \newrobustcmd*{#5}{\@ifstar{\csuse{@sgls@user@#1}}{\csuse{@gls@user@#1}}}%
1812 \expandafter\newcommand\expandafter*\expandafter
1813 {\csname @sgls@user@#1\endcsname}[1][{%
1814 \csuse{@gls@user@#1}[hyper=false,##1]%
1815 }%
1816 \expandafter\newcommand\expandafter*\expandafter
1817 {\csname @gls@user@#1\endcsname}[2][{%
1818 \new@ifnextchar[%
1819 {\csuse{@gls@user@#1@}{##1}{##2}}%
1820 {\csuse{@gls@user@#1@}{##1}{##2}[]}}%
1821 \csdef{@gls@user@#1@}##1##2[##3]{%
1822 \@gls@field@link{##1}{##2}{#3{##2}##3}%
1823 }%
1824 }%
```

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

```
1825 \ifcsdef{@Gls@user@#1@}%
1826 {%
1827 \PackageError{glossaries}%
1828 {Can't define '\string#6' as helper command
1829 '\expandafter\string\csname @Gls@user@#1@\endcsname' already exists}%
1830 }%
1831 }%
1832 {%
1833 \newrobustcmd*{#6}{\@ifstar{\csuse{@sGls@user@#1}}{\csuse{@Gls@user@#1}}}%
1834 \expandafter\newcommand\expandafter*\expandafter
1835 {\csname @sGls@user@#1\endcsname}[1][{%
1836 \csuse{@Gls@user@#1}[hyper=false,##1]%
1837 }%
1838 }
```

```

1838 \expandafter\newcommand\expandafter*\expandafter
1839 {\csname @Gls@user@#1\endcsname}[2][]{%
1840 \new@ifnextchar[%
1841 {\csuse{@Gls@user@#1@}{##1}{##2}}}%
1842 {\csuse{@Gls@user@#1@}{##1}{##2}[]}}}%
1843 \csdef{@Gls@user@#1@}##1##2[##3]{%
1844 \@gls@field@link{##1}{##2}{#4{##2}##3}%
1845 }%
1846 }%

```

Finally the all caps version:

```

1847 \ifcsdef{@GLS@user@#1@}%
1848 {%
1849 \PackageError{glossaries}%
1850 {Can't define '\string#7' as helper command
1851 '\expandafter\string\csname @GLS@user@#1@\endcsname' already exists}%
1852 }%
1853 }%
1854 {%
1855 \newrobustcmd*{#7}{\@ifstar{\csuse{@sGLS@user@#1}}{\csuse{@GLS@user@#1}}}%
1856 \expandafter\newcommand\expandafter*\expandafter
1857 {\csname @sGLS@user@#1\endcsname}[1][]{%
1858 \csuse{@GLS@user@#1}[hyper=false,##1]%
1859 }%
1860 \expandafter\newcommand\expandafter*\expandafter
1861 {\csname @GLS@user@#1\endcsname}[2][]{%
1862 \new@ifnextchar[%
1863 {\csuse{@GLS@user@#1@}{##1}{##2}}}%
1864 {\csuse{@GLS@user@#1@}{##1}{##2}[]}}}%
1865 \csdef{@GLS@user@#1@}##1##2[##3]{%
1866 \@gls@field@link{##1}{##2}{\mfirstucMakeUppercase{#3{##2}##3}}}%
1867 }%
1868 }%
1869 }%
1870 {%
1871 \PackageError{glossaries}{Key '#1' already exists}{}%
1872 }%
1873 }

```

\glswritedefhook

```

1874 \newcommand*{\glswritedefhook}{}

```

\gls@assign@desc

```

1875 \newcommand*{\gls@assign@desc}[1]{%
1876 \gls@assign@field{#1}{desc}{\@glo@desc}%
1877 \gls@assign@field{\@glo@desc}{#1}{descplural}{\@glo@descplural}%
1878 }

```

ongnewglossaryentry

```

1879 \newcommand{\longnewglossaryentry}[3]{%
1880   \glsdoifnoexists{#1}%
1881   {%
1882     \bgroup
1883     \let\@org@newglossaryentryprehook\@newglossaryentryprehook
1884     \long\def\@newglossaryentryprehook{%
1885       \long\def\@glo@desc{#3\leavevmode\unskip\nopostdesc}%
1886       \@org@newglossaryentryprehook
1887     }%
1888     \renewcommand*\@gls@assign@desc[1]{%
1889       \global\cslet{glo@\glsdetoklabel{#1}@desc}{\@glo@desc}%
1890       \global\cslet{glo@\glsdetoklabel{#1}@descplural}{\@glo@desc}%
1891     }
1892     \gls@defglossaryentry{#1}{#2}%
1893   \egroup
1894 }
1895 }

```

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

```

1896 \@onlypreamble{\longnewglossaryentry}

```

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

```

1897 \newcommand{\longprovideglossaryentry}[3]{%
1898   \ifglsentryexists{#1}{}%
1899   {\longnewglossaryentry{#1}{#2}{#3}}%
1900 }
1901 \@onlypreamble{\longprovideglossaryentry}

```

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

Defines a new entry without checking if it already exists.

```

1902 \newcommand{\gls@defglossaryentry}[2]{%

```

Store label

```

1903   \edef\@glo@label{\glsdetoklabel{#1}}%

```

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

```

1904   \let\glslabel\@glo@label

```

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

```

1905   \let\@glo@name\@glsnoname

```

```

1906   \let\@glo@desc\@glsnodesc

```

```

1907   \let\@glo@descplural\@gls@default@value

```

```

1908   \let\@glo@type\@gls@default@value

```

```

1909   \let\@glo@symbol\@gls@default@value

```

```

1910 \let\@glo@symbolplural\@gls@default@value
1911 \let\@glo@text\@gls@default@value
1912 \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.)

```

1913 \let\@glo@first\@gls@default@value
1914 \let\@glo@firstplural\@gls@default@value

```

Set the default sort:

```

1915 \let\@glo@sort\@gls@default@value

```

Set the default counter:

```

1916 \let\@glo@counter\@gls@default@value
1917 \def\@glo@see{}%
1918 \def\@glo@parent{}%
1919 \def\@glo@prefix{}%
1920 \def\@glo@useri{}%
1921 \def\@glo@userii{}%
1922 \def\@glo@useriii{}%
1923 \def\@glo@useriv{}%
1924 \def\@glo@userv{}%
1925 \def\@glo@uservi{}%
1926 \def\@glo@short{}%
1927 \def\@glo@shortpl{}%
1928 \def\@glo@long{}%
1929 \def\@glo@longpl{}%

```

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

```

1930 \@newglossaryentryprehook

```

Extract key-val information from third parameter:

```

1931 \setkeys{glossentry}{#2}%

```

Check there is a default glossary.

```

1932 \ifundef\glsdefaultttype
1933 {%
1934   \PackageError{glossaries}%
1935     {No default glossary type (have you used ‘nomain’?)}%
1936     {If you use package option ‘nomain’ you must define
1937      a new glossary before you can define entries}%
1938 }%
1939 {}%

```

Assign type. This must be fully expandable

```
1940 \gls@assign@field{\glsdefaulttype}{\@glo@label}{type}{\@glo@type}%
1941 \edef\@glo@type{\glsentrytype{\@glo@label}}%
```

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

```
1942 \ifcsundef{glolist@\@glo@type}%
1943 {%
1944   \PackageError{glossaries}%
1945     {Glossary type ‘\@glo@type’ has not been defined}%
1946     {You need to define a new glossary type, before making entries
1947       in it}%
1948 }%
1949 {%
1950   \protected@edef\@glolist@{\csname glolist@\@glo@type\endcsname}%
1951   \expandafter\xdef\csname glolist@\@glo@type\endcsname{%
1952     \@glolist@{\@glo@label},}%
1953 }%
```

Initialise level to 0.

```
1954 \gls@level=0\relax
```

Has this entry been assigned a parent?

```
1955 \ifx\@glo@parent\@empty
```

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

```
1956 \expandafter\gdef\csname glo@\@glo@label @parent\endcsname{}%
1957 \else
```

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

```
1958 \ifdefequal\@glo@label\@glo@parent%
1959 {%
1960   \PackageError{glossaries}{Entry ‘\@glo@label’ can’t be its own parent}{}%
1961   \def\@glo@parent{}%
1962   \expandafter\gdef\csname glo@\@glo@label @parent\endcsname{}%
1963 }%
1964 {%
```

Check the parent exists:

```
1965 \ifglentryexists{\@glo@parent}%
1966 {%
```

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

```
1967 \expandafter\xdef\csname glo@\@glo@label @parent\endcsname{%
1968   \@glo@parent}%
```

Determine level.

```
1969 \gls@level=\csname glo@\@glo@parent @level\endcsname\relax
1970 \advance\gls@level by 1\relax
```

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

```
1971 \ifx\@glo@name\@glsnoname
1972 \expandafter\let\expandafter\@glo@name
```

```

1973         \csname glo@\@glo@parent @name\endcsname
    If name and plural haven't been specified, use same as the parent
1974         \ifx\@glo@plural\@gls@default@value
1975         \expandafter\let\expandafter\@glo@plural
1976         \csname glo@\@glo@parent @plural\endcsname
1977     \fi
1978 \fi
1979 }%
1980 {%

```

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

```

1981     \PackageError{glossaries}%
1982     {%
1983         Invalid parent '\@glo@parent'
1984         for entry '\@glo@label' - parent doesn't exist%
1985     }%
1986     {%
1987         Parent entries must be defined before their children%
1988     }%
1989     \def\@glo@parent{%
1990     \expandafter\gdef\csname glo@\@glo@label @parent\endcsname{%
1991     }%
1992     }%
1993 \fi

```

Set the level for this entry

```

1994     \expandafter\xdef\csname glo@\@glo@label @level\endcsname{\number\@gls@level}%

```

Define commands associated with this entry:

```

1995     \gls@assign@field{\@glo@name}{\@glo@label}{sortvalue}{\@glo@sort}%
1996     \letcs\@glo@sort{glo@\@glo@label @sortvalue}%
1997     \gls@assign@field{\@glo@name}{\@glo@label}{text}{\@glo@text}%
1998     \expandafter\gls@assign@field\expandafter
1999         {\csname glo@\@glo@label @text\endcsname\glspluralsuffix}%
2000         {\@glo@label}{plural}{\@glo@plural}%
2001     \expandafter\gls@assign@field\expandafter
2002         {\csname glo@\@glo@label @text\endcsname}%
2003         {\@glo@label}{first}{\@glo@first}%

```

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

```

2004     \ifx\@glo@first\@gls@default@value
2005     \expandafter\gls@assign@field\expandafter
2006         {\csname glo@\@glo@label @plural\endcsname}%
2007         {\@glo@label}{firstpl}{\@glo@firstplural}%
2008     \else
2009     \expandafter\gls@assign@field\expandafter
2010         {\csname glo@\@glo@label @first\endcsname\glspluralsuffix}%
2011         {\@glo@label}{firstpl}{\@glo@firstplural}%

```

```

2012 \fi
2013 \ifcsundef{@glo@type @counter}%
2014 {%
2015 \def@glo@defaultcounter{glscounter}%
2016 }%
2017 {%
2018 \letcs@glo@defaultcounter{@glo@type @counter}%
2019 }%
2020 \gls@assign@field{@glo@defaultcounter}{@glo@label}{counter}{@glo@counter}%
2021 \gls@assign@field{}{@glo@label}{useri}{@glo@useri}%
2022 \gls@assign@field{}{@glo@label}{userii}{@glo@userii}%
2023 \gls@assign@field{}{@glo@label}{useriii}{@glo@useriii}%
2024 \gls@assign@field{}{@glo@label}{useriv}{@glo@useriv}%
2025 \gls@assign@field{}{@glo@label}{userv}{@glo@userv}%
2026 \gls@assign@field{}{@glo@label}{uservi}{@glo@uservi}%
2027 \gls@assign@field{}{@glo@label}{short}{@glo@short}%
2028 \gls@assign@field{}{@glo@label}{shortpl}{@glo@shortpl}%
2029 \gls@assign@field{}{@glo@label}{long}{@glo@long}%
2030 \gls@assign@field{}{@glo@label}{longpl}{@glo@longpl}%
2031 \ifx@glo@name@glsnoname
2032 \glsnoname
2033 \let@glo@name@gls@default@value
2034 \fi
2035 \gls@assign@field{}{@glo@label}{name}{@glo@name}%

```

Set default numberlist if not defined:

```

2036 \ifcsundef{glo@@glo@label @numberlist}%
2037 {%
2038 \csxdef{glo@@glo@label @numberlist}{%
2039 \noexpand@gls@missingnumberlist{@glo@label}}%
2040 }%
2041 {}%

```

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.

```

2042 \def@glo@@desc{@glo@first}%
2043 \ifx@glo@desc@glo@@desc
2044 \let@glo@desc@glo@first
2045 \fi
2046 \ifx@glo@desc@glsnodesc
2047 \glsnodesc
2048 \let@glo@desc@gls@default@value
2049 \fi
2050 \gls@assign@desc{@glo@label}%

```

Set the sort key for this entry:

```

2051 \@gls@defsort{@glo@type}{@glo@label}%
2052 \def@glo@@symbol{@glo@text}%

```



```

2053 \ifx\@glo@symbol\@glo@@symbol
2054 \let\@glo@symbol\@glo@text
2055 \fi
2056 \gls@assign@field{\relax}{\@glo@label}{symbol}{\@glo@symbol}%
2057 \expandafter
2058 \gls@assign@field\expandafter
2059 {\csname glo@\@glo@label @symbol\endcsname}
2060 {\@glo@label}{symbolplural}{\@glo@symbolplural}%

```

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

```

2061 \expandafter\xdef\csname glo@\@glo@label @flagfalse\endcsname{%
2062 \noexpand\global
2063 \noexpand\let\expandafter\noexpand
2064 \csname ifglo@\@glo@label @flag\endcsname\noexpand\iffalse
2065 }%
2066 \expandafter\xdef\csname glo@\@glo@label @flagtrue\endcsname{%
2067 \noexpand\global
2068 \noexpand\let\expandafter\noexpand
2069 \csname ifglo@\@glo@label @flag\endcsname\noexpand\iftrue
2070 }%
2071 \csname glo@\@glo@label @flagfalse\endcsname

```

Sort out any cross-referencing if required.

```

2072 \ifdefined\@glo@see
2073 {}%
2074 {}%
2075 \protected@edef\@do@glsee{%
2076 \noexpand\@gls@fixbraces\noexpand\@glo@list\@glo@see
2077 \noexpand\@nil
2078 \noexpand\expandafter\noexpand\@glsee\noexpand\@glo@list{\@glo@label}}%
2079 \@do@glsee
2080 }%

```

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

```

2081 \do@glo@storeentry{\@glo@label}%

```

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

```

2082 \@newglossaryentryposthook
2083 }

```

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

```

2084 \newcommand*{\@newglossaryentryprehook}{}

```

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

```

2085 \newcommand*{\@newglossaryentryposthook}{}

```

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

```

2086 \newcommand*{\glsmoveentry}[2]{%

```

```

2087 \edef\@glo@thislabel{\glstoklabel{#1}}%
2088 \edef\glo@type{\csname glo@\@glo@thislabel @type\endcsname}%
2089 \def\glo@list{,}%
2090 \forlslentries[\glo@type]{\glo@label}%
2091   {%
2092     \ifdefequal\@glo@thislabel\glo@label
2093       {\eappto\glo@list{\glo@label,}}%
2094     }%
2095     \cslet{glolist@\glo@type}{\glo@list}%
2096     \csdef{glo@\@glo@thislabel @type}{#2}%
2097 }

```

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

```

2098 \ifglxindy
2099   \newcommand*{\@glossaryentryfield}{\string\glossentry}
2100 \else
2101   \newcommand*{\@glossaryentryfield}{\string\glossentry}
2102 \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.)

```

2103 \ifglxindy
2104   \newcommand*{\@glossarysubentryfield}{%
2105     \string\subglossentry}
2106 \else
2107   \newcommand*{\@glossarysubentryfield}{%
2108     \string\subglossentry}
2109 \fi

```

\@glo@storeentry `\@glo@storeentry{<label>}`

Determine the format to write the entry in the glossary output (.glo) file. The argument is the entry's label (should already have been de-tok'ed if required). The result is stored in `\glo@<label>@entry`, where `<label>` is the entry's label. (This doesn't include any formatting or location information.)

```

2110 \newcommand{\@glo@storeentry}[1]{%

```

Escape makeindex/xindy special characters in the label:

```

2111 \edef\@glo@esclabel{#1}%
2112 \@gls@checkmkidxchars\@glo@esclabel

```

Get the sort string and escape any special characters

```

2113 \protected@edef\@glo@sort{\csname glo@#1@sort\endcsname}%
2114 \@gls@checkmkidxchars\@glo@sort

```

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

```

2115 \@gls@checkmkidxchars\@glo@prefix

Get the parent, if one exists
2116 \edef\@glo@parent{\csname glo@#1@parent\endcsname}%

Write the information to the glossary file.
2117 \ifglxsindy

Store using xindy syntax.
2118 \ifx\@glo@parent\@empty

Entry doesn't have a parent
2119 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
2120 (\string"\@glo@sort\string" %
2121 \string"\@glo@prefix\@glossaryentryfield{\@glo@esclabel}\string") %
2122 }%
2123 \else

Entry has a parent
2124 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
2125 \csname glo@\@glo@parent @index\endcsname
2126 (\string"\@glo@sort\string" %
2127 \string"\@glo@prefix\@glossarysubentryfield
2128 {\csname glo@#1@level\endcsname}{\@glo@esclabel}\string") %
2129 }%
2130 \fi
2131 \else

Store using makeindex syntax.
2132 \ifx\@glo@parent\@empty

Sanitize \@glo@prefix
2133 \@onelevel@sanitize\@glo@prefix

Entry doesn't have a parent
2134 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
2135 \@glo@sort\@gls@actualchar\@glo@prefix
2136 \@glossaryentryfield{\@glo@esclabel}%
2137 }%
2138 \else

Entry has a parent
2139 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
2140 \csname glo@\@glo@parent @index\endcsname\@gls@levelchar
2141 \@glo@sort\@gls@actualchar\@glo@prefix
2142 \@glossarysubentryfield
2143 {\csname glo@#1@level\endcsname}{\@glo@esclabel}%
2144 }%
2145 \fi
2146 \fi
2147 }

```

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`

```
2148 \AtBeginDocument{%
2149   \ifpackageloaded{amsmath}%
2150   {\let\gls@ifnotmeasuring\@gls@ifnotmeasuring}%
2151   }%
2152 }
2153 \newcommand*{\@gls@ifnotmeasuring}[1]{%
2154   \ifmeasuring@
2155   \else
2156     #1%
2157   \fi
2158 }
2159 \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.

```
2160 \newcommand*{\glsreset}[1]{%
2161   \gls@ifnotmeasuring
2162   {%
2163     \glsdoifexists{#1}%
2164     {%
2165       \expandafter\global\csname glo@\glsdetoklabel{#1}@flagfalse\endcsname
2166     }%
2167   }%
2168 }
```

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

```
2169 \newcommand*{\glslocalreset}[1]{%
2170   \gls@ifnotmeasuring
2171   {%
2172     \glsdoifexists{#1}%
2173     {%
2174       \expandafter\let\csname ifglo@\glsdetoklabel{#1}@flag\endcsname\iffalse
2175     }%
2176   }%
2177 }
```

`\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.

```
2178 \newcommand*{\glsunset}[1]{%
```

```

2179 \gls@ifnotmeasuring
2180 {%
2181     \glsdoifexists{#1}%
2182     {%
2183         \expandafter\global\csname glo@\glsdetoklabel{#1}@flagtrue\endcsname
2184     }%
2185 }%
2186 }

```

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

```

2187 \newcommand*\glslocalunset}[1]{%
2188     \gls@ifnotmeasuring
2189     {%
2190         \glsdoifexists{#1}%
2191         {%
2192             \expandafter\let\csname ifglo@\glsdetoklabel{#1}@flag\endcsname\iftrue
2193         }%
2194     }%
2195 }

```

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

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

`\glsresetall`

```

2196 \newcommand*\glsresetall}[1][\@glo@types]{%
2197     \forallglsentries[#1]{\@glsentry}%
2198     {%
2199         \glsreset{\@glsentry}%
2200     }%
2201 }

```

As above, but with only a local effect:

`\glslocalresetall`

```

2202 \newcommand*\glslocalresetall}[1][\@glo@types]{%
2203     \forallglsentries[#1]{\@glsentry}%
2204     {%
2205         \glslocalreset{\@glsentry}%
2206     }%
2207 }

```

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

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

`\glsunsetall`

```

2208 \newcommand*\glsunsetall}[1][\@glo@types]{%
2209     \forallglsentries[#1]{\@glsentry}%
2210     {%
2211         \glsunset{\@glsentry}%
2212     }%
2213 }

```

As above, but with only a local effect:

```
\glslocalunsetall
```

```
2214 \newcommand*{\glslocalunsetall}[1][\@gls@types]{%
2215   \forallglsentries[#1]{\@glsentry}%
2216   {%
2217     \glslocalunset{\@glsentry}%
2218   }%
2219 }
```

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
```

```
2220 \newcommand*{\loadglsentries}[2][\@gls@default]{%
2221   \let\@gls@default\glsdefaulttype
2222   \def\glsdefaulttype{#1}\input{#2}%
2223   \let\glsdefaulttype\@gls@default
2224 }
```

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

```
2225 \@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
```

```
2226 \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

```
2227 \newcommand*{\glsentryfmt}{%
2228   \@@gls@default@entryfmt\glsdisplayfirst\glsdisplay
2229 }
```

Format that provides backwards compatibility:

```
2230 \newcommand*{\@@gls@default@entryfmt}[2]{%
2231   \ifdefempty\glscustomtext
2232     {%
2233       \glsifplural
2234       {%
```

Plural form

```
2235       \glscapscase
2236       {%
```

Don't adjust case

```
2237       \ifglsused\glslabel
2238       {%
```

Subsequent use

```
2239         #2{\glsentryplural{\glslabel}}%
2240         {\glsentrydescplural{\glslabel}}%
2241         {\glsentrysymbolplural{\glslabel}}{\glsinsert}%
2242     }%
2243     {%
```

First use

```
2244         #1{\glsentryfirstplural{\glslabel}}%
2245         {\glsentrydescplural{\glslabel}}%
2246         {\glsentrysymbolplural{\glslabel}}{\glsinsert}%
2247     }%
2248 }%
2249 {%
```

Make first letter upper case

```
2250     \ifglsused\glslabel
2251     {%
```

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.)

```
2252     \ifbool{glscompatible-3.07}%
2253     {%
2254       \protected@edef\@glo@etext{%
2255         #2{\glsentryplural{\glslabel}}%
2256         {\glsentrydescplural{\glslabel}}%
```

```

2257         {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2258     \xmakefirstuc\@glo@etext
2259 }%
2260 {%
2261     #2{\Glsentryplural{\glslabel}}%
2262     {\glsentrydescplural{\glslabel}}%
2263     {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2264 }%
2265 }%
2266 {%

```

First use

```

2267     \ifbool{glscompatible-3.07}%
2268     {%
2269         \protected@edef\@glo@etext{%
2270             #1{\glsentryfirstplural{\glslabel}}%
2271             {\glsentrydescplural{\glslabel}}%
2272             {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2273         \xmakefirstuc\@glo@etext
2274     }%
2275     {%
2276         #1{\Glsentryfirstplural{\glslabel}}%
2277         {\glsentrydescplural{\glslabel}}%
2278         {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2279     }%
2280 }%
2281 }%
2282 {%

```

Make all upper case

```

2283     \ifglsused\glslabel
2284     {%

```

Subsequent use

```

2285         \mfirstucMakeUppercase{#2{\glsentryplural{\glslabel}}%
2286         {\glsentrydescplural{\glslabel}}%
2287         {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2288     }%
2289     {%

```

First use

```

2290         \mfirstucMakeUppercase{#1{\glsentryfirstplural{\glslabel}}%
2291         {\glsentrydescplural{\glslabel}}%
2292         {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2293     }%
2294 }%
2295 }%
2296 {%

```

Singular form

```

2297     \glscapscase
2298     {%

```


Don't adjust case

```
2299      \ifglsused\glslabel
2300      {%
```

Subsequent use

```
2301      #2{\glsentrytext{\glslabel}}%
2302      {\glsentrydesc{\glslabel}}%
2303      {\glsentrysymbol{\glslabel}}{\glsinsert}%
2304      }%
2305      {%
```

First use

```
2306      #1{\glsentryfirst{\glslabel}}%
2307      {\glsentrydesc{\glslabel}}%
2308      {\glsentrysymbol{\glslabel}}{\glsinsert}%
2309      }%
2310      }%
2311      {%
```

Make first letter upper case

```
2312      \ifglsused\glslabel
2313      {%
```

Subsequent use

```
2314      \ifbool{glscompatible-3.07}%
2315      {%
2316      \protected@edef\@glo@etext{%
2317      #2{\glsentrytext{\glslabel}}%
2318      {\glsentrydesc{\glslabel}}%
2319      {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2320      \xmakefirstuc\@glo@etext
2321      }%
2322      {%
2323      #2{\Glsentrytext{\glslabel}}%
2324      {\glsentrydesc{\glslabel}}%
2325      {\glsentrysymbol{\glslabel}}{\glsinsert}%
2326      }%
2327      }%
2328      {%
```

First use

```
2329      \ifbool{glscompatible-3.07}%
2330      {%
2331      \protected@edef\@glo@etext{%
2332      #1{\glsentryfirst{\glslabel}}%
2333      {\glsentrydesc{\glslabel}}%
2334      {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2335      \xmakefirstuc\@glo@etext
2336      }%
2337      {%
2338      #1{\Glsentryfirst{\glslabel}}%
```

```

2339          {\glsentrydesc{\glslabel}}%
2340          {\glsentrysymbol{\glslabel}}{\glsinsert}%
2341      }%
2342  }%
2343  }%
2344  {%

  Make all upper case
2345      \ifglsused\glslabel
2346      {%

    Subsequent use
2347        \mfirstucMakeUppercase{#2{\glsentrytext{\glslabel}}%
2348        {\glsentrydesc{\glslabel}}%
2349        {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2350    }%
2351    {%

    First use
2352        \mfirstucMakeUppercase{#1{\glsentryfirst{\glslabel}}%
2353        {\glsentrydesc{\glslabel}}%
2354        {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2355    }%
2356  }%
2357  }%
2358  }%
2359  {%

  Custom text provided in \glsdisp
2360      \ifglsused{\glslabel}%
2361      {%

    Subsequent use
2362        #2{\glscustomtext}%
2363        {\glsentrydesc{\glslabel}}%
2364        {\glsentrysymbol{\glslabel}}{%
2365    }%
2366    {%

    First use
2367        #1{\glscustomtext}%
2368        {\glsentrydesc{\glslabel}}%
2369        {\glsentrysymbol{\glslabel}}{%
2370    }%
2371    }%
2372 }

```

`\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.

```

2373 \newcommand*{\glsgenentryfmt}{%
2374   \ifdefempty\glscustomtext

```

```

2375  {%
2376    \glsifplural
2377  {%

```

Plural form

```

2378    \glscapscase
2379  {%

```

Don't adjust case

```

2380    \ifglsused\glslabel
2381  {%

```

Subsequent use

```

2382    \glsentryplural{\glslabel}\glsinsert
2383  }%
2384  {%

```

First use

```

2385    \glsentryfirstplural{\glslabel}\glsinsert
2386  }%
2387  }%
2388  {%

```

Make first letter upper case

```

2389    \ifglsused\glslabel
2390  {%

```

Subsequent use.

```

2391    \Glsentryplural{\glslabel}\glsinsert
2392  }%
2393  {%

```

First use

```

2394    \Glsentryfirstplural{\glslabel}\glsinsert
2395  }%
2396  }%
2397  {%

```

Make all upper case

```

2398    \ifglsused\glslabel
2399  {%

```

Subsequent use

```

2400    \mfirstucMakeUppercase
2401    {\glsentryplural{\glslabel}\glsinsert}%
2402  }%
2403  {%

```

First use

```

2404    \mfirstucMakeUppercase
2405    {\glsentryfirstplural{\glslabel}\glsinsert}%
2406  }%
2407  }%

```

2408 }%

2409 {%

Singular form

2410 \glscapscase

2411 {%

Don't adjust case

2412 \ifglused\glslabel

2413 {%

Subsequent use

2414 \glstrytext{\glslabel}\glinsert

2415 }%

2416 {%

First use

2417 \glstryfirst{\glslabel}\glinsert

2418 }%

2419 }%

2420 {%

Make first letter upper case

2421 \ifglused\glslabel

2422 {%

Subsequent use

2423 \Glstrytext{\glslabel}\glinsert

2424 }%

2425 {%

First use

2426 \Glstryfirst{\glslabel}\glinsert

2427 }%

2428 }%

2429 {%

Make all upper case

2430 \ifglused\glslabel

2431 {%

Subsequent use

2432 \mfirstucMakeUppercase{\glstrytext{\glslabel}\glinsert}%

2433 }%

2434 {%

First use

2435 \mfirstucMakeUppercase{\glstryfirst{\glslabel}\glinsert}%

2436 }%

2437 }%

2438 }%

2439 }%

2440 {%

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

```
2441 \glscustomtext\glsinsert
2442 }%
2443 }
```

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

```
2444 \newcommand*{\glsngenacfmt}{%
2445 \ifdefempty\glscustomtext
2446 {%
2447 \ifglsused\glslabel
2448 {%
```

Subsequent use:

```
2449 \glsifplural
2450 {%
```

Subsequent plural form:

```
2451 \glscapscase
2452 {%
```

Subsequent plural form, don't adjust case:

```
2453 \acronymfont{\glsentryshortpl{\glslabel}}\glsinsert
2454 }%
2455 {%
```

Subsequent plural form, make first letter upper case:

```
2456 \acronymfont{\Glsentryshortpl{\glslabel}}\glsinsert
2457 }%
2458 {%
```

Subsequent plural form, all caps:

```
2459 \mfirstucMakeUppercase
2460 {\acronymfont{\glsentryshortpl{\glslabel}}\glsinsert}%
2461 }%
2462 }%
2463 {%
```

Subsequent singular form

```
2464 \glscapscase
2465 {%
```

Subsequent singular form, don't adjust case:

```
2466 \acronymfont{\glsentryshort{\glslabel}}\glsinsert
2467 }%
2468 {%
```

Subsequent singular form, make first letter upper case:

```
2469 \acronymfont{\Glsentryshort{\glslabel}}\glsinsert
2470 }%
2471 {%
```

Subsequent singular form, all caps:

```
2472      \mfirstucMakeUppercase
2473      {\acronymfont{\glsentryshort{\glslabel}}\glsinsert}%
2474      }%
2475      }%
2476      }%
2477      {%
```

First use:

```
2478      \glsifplural
2479      {%
```

First use plural form:

```
2480      \glscapscase
2481      {%
```

First use plural form, don't adjust case:

```
2482      \genplacrfullformat{\glslabel}{\glsinsert}%
2483      }%
2484      {%
```

First use plural form, make first letter upper case:

```
2485      \Genplacrfullformat{\glslabel}{\glsinsert}%
2486      }%
2487      {%
```

First use plural form, all caps:

```
2488      \mfirstucMakeUppercase
2489      {\genplacrfullformat{\glslabel}{\glsinsert}}%
2490      }%
2491      }%
2492      {%
```

First use singular form

```
2493      \glscapscase
2494      {%
```

First use singular form, don't adjust case:

```
2495      \genacrfullformat{\glslabel}{\glsinsert}%
2496      }%
2497      {%
```

First use singular form, make first letter upper case:

```
2498      \Genacrfullformat{\glslabel}{\glsinsert}%
2499      }%
2500      {%
```

First use singular form, all caps:

```
2501      \mfirstucMakeUppercase
2502      {\genacrfullformat{\glslabel}{\glsinsert}}%
2503      }%
2504      }%
2505      }%
```

```

2506 }%
2507 {%
    User supplied text.
2508 \glscustomtext
2509 }%
2510 }

```

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

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

```

2511 \newcommand*{\genacrfullformat}[2]{%
2512   \glentrylong{#1}#2\space
2513   (\protect\firstacronymfont{\glentryshort{#1}})%
2514 }

```

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

As above but makes the first letter upper case.

```

2515 \newcommand*{\Genacrfullformat}[2]{%
2516   \protected@edef\gls@text{\genacrfullformat{#1}{#2}}%
2517   \xmakefirstuc\gls@text
2518 }

```

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

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

```

2519 \newcommand*{\genplacrfullformat}[2]{%
2520   \glentrylongpl{#1}#2\space
2521   (\protect\firstacronymfont{\glentryshortpl{#1}})%
2522 }

```

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

As above but makes the first letter upper case.

```

2523 \newcommand*{\Genplacrfullformat}[2]{%
2524   \protected@edef\gls@text{\genplacrfullformat{#1}{#2}}%
2525   \xmakefirstuc\gls@text
2526 }

```

`\glsdisplayfirst` Deprecated. Kept for backward compatibility.

```

2527 \newcommand*{\glsdisplayfirst}[4]{#1#4}

```

`\glsdisplay` Deprecated. Kept for backward compatibility.

```
2528 \newcommand*{\glsdisplay}[4]{#1#4}
```

`\defglsdisplay` Deprecated. Kept for backward compatibility.

```
2529 \newcommand*{\defglsdisplay}[2][\glsdefaulttype]{%
2530   \GlossariesWarning{\string\defglsdisplay\space is now obsolete.^^J
2531   Use \string\defglsentryfmt\space instead}%
2532   \expandafter\def\csname gls@#1@display\endcsname##1##2##3##4{#2}%
2533   \edef\@gls@doentrydef{%
2534     \noexpand\defglsentryfmt[#1]{%
2535       \noexpand\ifcsdef{gls@#1@displayfirst}%
2536       {%
2537         \noexpand\@@gls@default@entryfmt
2538         {\noexpand\csuse{gls@#1@displayfirst}}}%
2539       {\noexpand\csuse{gls@#1@display}}}%
2540     }%
2541     {%
2542       \noexpand\@@gls@default@entryfmt
2543       {\noexpand\glsdisplayfirst}%
2544       {\noexpand\csuse{gls@#1@display}}}%
2545     }%
2546   }%
2547 }%
2548 \@gls@doentrydef
2549 }
```

`\defglsdisplayfirst` Deprecated. Kept for backward compatibility.

```
2550 \newcommand*{\defglsdisplayfirst}[2][\glsdefaulttype]{%
2551   \GlossariesWarning{\string\defglsdisplayfirst\space is now obsolete.^^J
2552   Use \string\defglsentryfmt\space instead}%
2553   \expandafter\def\csname gls@#1@displayfirst\endcsname##1##2##3##4{#2}%
2554   \edef\@gls@doentrydef{%
2555     \noexpand\defglsentryfmt[#1]{%
2556       \noexpand\ifcsdef{gls@#1@display}%
2557       {%
2558         \noexpand\@@gls@default@entryfmt
2559         {\noexpand\csuse{gls@#1@displayfirst}}}%
2560       {\noexpand\csuse{gls@#1@display}}}%
2561     }%
2562     {%
2563       \noexpand\@@gls@default@entryfmt
2564       {\noexpand\csuse{gls@#1@displayfirst}}}%
2565       {\noexpand\glsdisplay}%
2566     }%
2567   }%
2568 }%
2569 \@gls@doentrydef
2570 }
```


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.

```
2571 \define@key{glslink}{counter}{%
2572   \ifcsundef{c@#1}%
2573   {%
2574     \PackageError{glossaries}%
2575     {There is no counter called ‘#1’}%
2576     {%
2577       The counter key should have the name of a valid counter
2578       as its value%
2579     }%
2580   }%
2581   {%
2582     \def\@gls@counter{#1}%
2583   }%
2584 }
```

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.

```
2585 \define@key{glslink}{format}{%
2586   \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.

```
2587 \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.

```
2588 \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
2589 \newrobustcmd*{\glslink}{%
2590 \ifstar\@sgls@link\@gls@@link
2591 }

\@sgls@link The starred version of \glslink calls the unstarred version with hyperlinks dis-
abled.
2592 \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.
2593 \newcommand*{\@gls@@link}[3] [] {%
2594 \ifglsentryexists{#2}%
2595 {%
2596 \@gls@link[#1]{#2}{#3}%
2597 }{%
2598 \PackageError{glossaries}{Glossary entry '#2' has not been
2599 defined}{You need to define a glossary entry before you
2600 can use it.}%
2601 \gls@textformat{#3}%
2602 }%
2603 }

\@gls@link
2604 \def\@gls@link[#1]#2#3{%
Inserting \leavevmode suggested by Donald Arseneau (avoids problem with
tabularx).
2605 \leavevmode
2606 \edef\glslabel{\glsdetoklabel{#2}}%
Save options in \@gls@link@opts and label in \@gls@link@label
2607 \def\@gls@link@opts{#1}%
2608 \let\@gls@link@label\glslabel
```

```

2609 \def\@glsnumberformat{glsnumberformat}%
2610 \edef\@gls@counter{\csname glo@\glslabel @counter\endcsname}%

```

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

```

2611 \edef\gls@type{\csname glo@\glslabel @type\endcsname}%
2612 \expandafter\DTLifinlist\expandafter
2613   {\gls@type}{\@gls@nohyperlist}%
2614   {%
2615     \KV@glslink@hyperfalse
2616   }%
2617   {%
2618     \KV@glslink@hypertrue
2619   }%
2620 \setkeys{glslink}{#1}%

```

Store the entry’s counter in \theglsentrycounter

```

2621 \@gls@saveentrycounter

```

Define sort key if necessary:

```

2622 \@gls@setsort{\glslabel}%

```

(De-tok’ing done by \@do@wrglossary)

```

2623 \@do@wrglossary{#2}%
2624 \ifKV@glslink@hyper
2625   \@glslink{\glolinkprefix\glslabel}{\glstextformat{#3}}%
2626 \else
2627   \glstextformat{#3}%
2628 \fi
2629 }

```

\glolinkprefix

```

2630 \newcommand*{\glolinkprefix}{glo:}

```

\glsentrycounter Set default value of entry counter

```

2631 \def\glsentrycounter{\glscounter}%

```

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

```

2632 \newcommand*{\@gls@saveentrycounter}{%
2633 \def\@gls@Hcounter{}%

```

Are we using equation counter?

```

2634 \ifthenelse{\equal{\@gls@counter}{equation}}%
2635 {

```

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

```

2636 \ifcsundef{xatlevel@}%
2637 {%
2638 \edef\theglsentrycounter{\expandafter\noexpand

```

```

2639     \csname the\@gls@counter\endcsname}%
2640 }%
2641 {%
2642     \ifx\xatlevel@\@empty
2643         \edef\theglentrycounter{\expandafter\noexpand
2644             \csname the\@gls@counter\endcsname}%
2645     \else
2646         \savecounters@
2647         \advance\c@equation by 1\relax
2648         \edef\theglentrycounter{\csname the\@gls@counter\endcsname}%

```

Check if hyperref version of this counter

```

2649     \ifcsundef{theH\@gls@counter}%
2650     {%
2651         \def\@gls@Hcounter{\theglentrycounter}%
2652     }%
2653     {%
2654         \def\@gls@Hcounter{\csname theH\@gls@counter\endcsname}%
2655     }%
2656     \protected@edef\theHglentrycounter{\@gls@Hcounter}%
2657     \restorecounters@
2658 \fi
2659 }%
2660 }%
2661 {%

```

Not using equation counter so no special measures:

```

2662     \edef\theglentrycounter{\expandafter\noexpand
2663         \csname the\@gls@counter\endcsname}%
2664 }%

```

Check if hyperref version of this counter

```

2665 \ifx\@gls@Hcounter\@empty
2666 \ifcsundef{theH\@gls@counter}%
2667 {%
2668     \def\theHglentrycounter{\theglentrycounter}%
2669 }%
2670 {%
2671     \protected@edef\theHglentrycounter{\expandafter\noexpand
2672         \csname theH\@gls@counter\endcsname}%
2673 }%
2674 \fi
2675 }

```

`\@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.

```

2676 \def\@set@glo@numformat#1#2#3#4{%
2677   \expandafter\@glo@check@mkidxrangechar#3\@nil
2678   \protected@edef#1{%
2679     \@glo@prefix setentrycounter[#4]{#2}%
2680     \expandafter\string\csname\@glo@suffix\endcsname
2681   }%
2682   \@gls@checkmkidxchars#1%
2683 }

```

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.

```

2684 \def\@glo@check@mkidxrangechar#1#2\@nil{%
2685   \if#1(\relax
2686     \def\@glo@prefix{(%
2687     \if\relax#2\relax
2688       \def\@glo@suffix{glsnumberformat}%
2689     \else
2690       \def\@glo@suffix{#2}%
2691     \fi
2692   \else
2693     \if#1)\relax
2694       \def\@glo@prefix{)%
2695     \if\relax#2\relax
2696       \def\@glo@suffix{glsnumberformat}%
2697     \else
2698       \def\@glo@suffix{#2}%
2699     \fi
2700   \else
2701     \def\@glo@prefix{}\def\@glo@suffix{#1#2}%
2702   \fi
2703 \fi}

```

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

```

2704 \newcommand*\@gls@escbsdq[1]{%
2705   \def\@gls@checkedmkidx{}%
2706   \let\gls@xdystring=#1\relax
2707   \@onelevel@sanitize\gls@xdystring
2708   \edef\do@gls@xdycheckbackslash{%
2709     \noexpand\@gls@xdycheckbackslash\gls@xdystring\noexpand\@nil
2710     \@backslashchar\@backslashchar\noexpand\null}%
2711   \do@gls@xdycheckbackslash
2712   \expandafter\@gls@updatechecked\@gls@checkedmkidx{\gls@xdystring}%
2713   \def\@gls@checkedmkidx{}%
2714   \expandafter\@gls@xdycheckquote\gls@xdystring\@nil""\null
2715   \expandafter\@gls@updatechecked\@gls@checkedmkidx{\gls@xdystring}%

```

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

```

2716 \for\@gls@tmp:=\gls@protected@pagefmts\do
2717 {%
2718   \edef\@gls@sanitized@tmp{\expandafter\@gobble\string\\expandonce\@gls@tmp}%
2719   \@onelevel@sanitize\@gls@sanitized@tmp
2720   \edef\gls@dosubst{%
2721     \noexpand\DTLsubstituteall\noexpand\gls@xdyststring
2722     {\@gls@sanitized@tmp}{\expandonce\@gls@tmp}%
2723   }%
2724   \gls@dosubst
2725 }%

```

Assign to required control sequence

```

2726 \let#1=\gls@xdyststring
2727 }

```

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

\gls@checkmkidxchars

```

2728 \newcommand{\@gls@checkmkidxchars}[1]{%
2729   \ifglxsindy
2730     \@gls@escbsdq{#1}%
2731   \else
2732     \def\@gls@checkedmkidx{%
2733       \expandafter\@gls@checkquote#1\@nil""\null
2734       \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2735     \def\@gls@checkedmkidx{%
2736       \expandafter\@gls@checkescquote#1\@nil\\"\null
2737       \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2738     \def\@gls@checkedmkidx{%
2739       \expandafter\@gls@checkescactual#1\@nil\?\?\null
2740       \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2741     \def\@gls@checkedmkidx{%
2742       \expandafter\@gls@checkactual#1\@nil??\null
2743       \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2744     \def\@gls@checkedmkidx{%
2745       \expandafter\@gls@checkbar#1\@nil||\null
2746       \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2747     \def\@gls@checkedmkidx{%
2748       \expandafter\@gls@checkescbar#1\@nil\\|\null
2749       \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2750     \def\@gls@checkedmkidx{%
2751       \expandafter\@gls@checklevel#1\@nil!!\null
2752       \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2753     \fi
2754 }

```

Update the control sequence and strip trailing \@nil:

\@gls@updatechecked

```
2755 \def\@gls@updatechecked#1\@nil#2{\def#2{#1}}
```

\@gls@tmpb Define temporary token

```
2756 \newtoks\@gls@tmpb
```

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

```
2757 \def\@gls@checkquote#1"#2"#3\null{%
2758   \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2759   \toks@={#1}%
2760   \ifx\null#2\null
2761     \ifx\null#3\null
2762       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2763       \def\@gls@checkquote{\relax}%
2764     \else
2765       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2766         \@gls@quotechar\@gls@quotechar\@gls@quotechar\@gls@quotechar}%
2767       \def\@gls@checkquote{\@gls@checkquote#3\null}%
2768     \fi
2769   \else
2770     \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2771       \@gls@quotechar\@gls@quotechar}%
2772     \ifx\null#3\null
2773       \def\@gls@checkquote{\@gls@checkquote#2""\null}%
2774     \else
2775       \def\@gls@checkquote{\@gls@checkquote#2"#3\null}%
2776     \fi
2777   \fi
2778   \@gls@checkquote
2779 }
```

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

```
2780 \def\@gls@checkescquote#1\"#2\"#3\null{%
2781   \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2782   \toks@={#1}%
2783   \ifx\null#2\null
2784     \ifx\null#3\null
2785       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2786       \def\@gls@checkescquote{\relax}%
2787     \else
2788       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2789         \@gls@quotechar\string\" \@gls@quotechar
2790         \@gls@quotechar\string\" \@gls@quotechar}%
2791       \def\@gls@checkescquote{\@gls@checkescquote#3\null}%
2792     \fi
2793   \else
2794     \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2795       \@gls@quotechar\string\" \@gls@quotechar}%
2796     \ifx\null#3\null
```

```

2797     \def\@gls@checkescquote{\@gls@checkescquote#2\""\null}%
2798   \else
2799     \def\@gls@checkescquote{\@gls@checkescquote#2\"#3\null}%
2800   \fi
2801 \fi
2802 \@gls@checkescquote
2803 }

```

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

```

2804 \def\@gls@checkescactual#1\?#2\?#3\null{%
2805   \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2806   \toks@={#1}%
2807   \ifx\null#2\null
2808     \ifx\null#3\null
2809       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2810       \def\@gls@checkescactual{\relax}%
2811     \else
2812       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2813         \@gls@quotearchar\string\" \@gls@actualchar
2814         \@gls@quotearchar\string\" \@gls@actualchar}%
2815       \def\@gls@checkescactual{\@gls@checkescactual#3\null}%
2816     \fi
2817   \else
2818     \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2819       \@gls@quotearchar\string\" \@gls@actualchar}%
2820     \ifx\null#3\null
2821       \def\@gls@checkescactual{\@gls@checkescactual#2\?\?\null}%
2822     \else
2823       \def\@gls@checkescactual{\@gls@checkescactual#2\?#3\null}%
2824     \fi
2825   \fi
2826 \@gls@checkescactual
2827 }

```

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

```

2828 \def\@gls@checkescbar#1\|#2\|#3\null{%
2829   \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2830   \toks@={#1}%
2831   \ifx\null#2\null
2832     \ifx\null#3\null
2833       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2834       \def\@gls@checkescbar{\relax}%
2835     \else
2836       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2837         \@gls@quotearchar\string\" \@gls@encapchar
2838         \@gls@quotearchar\string\" \@gls@encapchar}%
2839       \def\@gls@checkescbar{\@gls@checkescbar#3\null}%
2840     \fi
2841   \else

```



```

2842 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2843 \@gls@quotearchar\string\"@gls@encapchar}%
2844 \ifx\null#3\null
2845 \def\@gls@checkescbar{\@gls@checkescbar#2\|\|\null}%
2846 \else
2847 \def\@gls@checkescbar{\@gls@checkescbar#2\|#3\null}%
2848 \fi
2849 \fi
2850 \@gls@checkescbar
2851 }

```

\@gls@checkesclevel Similarly for \!:

```

2852 \def\@gls@checkesclevel#1\!#2\!#3\null{%
2853 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2854 \toks@={#1}%
2855 \ifx\null#2\null
2856 \ifx\null#3\null
2857 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2858 \def\@gls@checkesclevel{\relax}%
2859 \else
2860 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2861 \@gls@quotearchar\string\"@gls@levelchar
2862 \@gls@quotearchar\string\"@gls@levelchar}%
2863 \def\@gls@checkesclevel{\@gls@checkesclevel#3\null}%
2864 \fi
2865 \else
2866 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2867 \@gls@quotearchar\string\"@gls@levelchar}%
2868 \ifx\null#3\null
2869 \def\@gls@checkesclevel{\@gls@checkesclevel#2\!\!\null}%
2870 \else
2871 \def\@gls@checkesclevel{\@gls@checkesclevel#2\!#3\null}%
2872 \fi
2873 \fi
2874 \@gls@checkesclevel
2875 }

```

\@gls@checkbar and for |:

```

2876 \def\@gls@checkbar#1|#2|#3\null{%
2877 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2878 \toks@={#1}%
2879 \ifx\null#2\null
2880 \ifx\null#3\null
2881 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2882 \def\@gls@checkbar{\relax}%
2883 \else
2884 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2885 \@gls@quotearchar\@gls@encapchar\@gls@quotearchar\@gls@encapchar}%
2886 \def\@gls@checkbar{\@gls@checkbar#3\null}%

```

```

2887 \fi
2888 \else
2889 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2890 \gls@quotechar\@gls@encapchar}%
2891 \ifx\null#3\null
2892 \def\@gls@checkbar{\@gls@checkbar#2||\null}%
2893 \else
2894 \def\@gls@checkbar{\@gls@checkbar#2|#3\null}%
2895 \fi
2896 \fi
2897 \@gls@checkbar
2898 }

```

\@gls@checklevel and for !:

```

2899 \def\@gls@checklevel#1!#2!#3\null{%
2900 \gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2901 \toks@={#1}%
2902 \ifx\null#2\null
2903 \ifx\null#3\null
2904 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2905 \def\@gls@checklevel{\relax}%
2906 \else
2907 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2908 \gls@quotechar\@gls@levelchar\@gls@quotechar\@gls@levelchar}%
2909 \def\@gls@checklevel{\@gls@checklevel#3\null}%
2910 \fi
2911 \else
2912 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2913 \gls@quotechar\@gls@levelchar}%
2914 \ifx\null#3\null
2915 \def\@gls@checklevel{\@gls@checklevel#2!!\null}%
2916 \else
2917 \def\@gls@checklevel{\@gls@checklevel#2!#3\null}%
2918 \fi
2919 \fi
2920 \@gls@checklevel
2921 }

```

\@gls@checkactual and for ?:

```

2922 \def\@gls@checkactual#1?#2?#3\null{%
2923 \gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2924 \toks@={#1}%
2925 \ifx\null#2\null
2926 \ifx\null#3\null
2927 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2928 \def\@gls@checkactual{\relax}%
2929 \else
2930 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2931 \gls@quotechar\@gls@actualchar\@gls@quotechar\@gls@actualchar}%

```

```

2932     \def\@gls@checkactual{\@gls@checkactual#3\null}%
2933     \fi
2934   \else
2935     \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2936       \@gls@quotechar\@gls@actualchar}%
2937     \ifx\null#3\null
2938       \def\@gls@checkactual{\@gls@checkactual#2??\null}%
2939     \else
2940       \def\@gls@checkactual{\@gls@checkactual#2?#3\null}%
2941     \fi
2942   \fi
2943   \@gls@checkactual
2944 }

```

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

```

2945 \def\@gls@xdycheckquote#1"#2"#3\null{%
2946   \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2947   \toks@={#1}%
2948   \ifx\null#2\null
2949     \ifx\null#3\null
2950       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2951       \def\@gls@xdycheckquote{\relax}%
2952     \else
2953       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2954         \string"\string"}%
2955       \def\@gls@xdycheckquote{\@gls@xdycheckquote#3\null}%
2956     \fi
2957   \else
2958     \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2959       \string"}%
2960     \ifx\null#3\null
2961       \def\@gls@xdycheckquote{\@gls@xdycheckquote#2""\null}%
2962     \else
2963       \def\@gls@xdycheckquote{\@gls@xdycheckquote#2"#3\null}%
2964     \fi
2965   \fi
2966   \@gls@xdycheckquote
2967 }

```

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

```

\@gls@xdycheckbackslash
2968 \edef\def\@gls@xdycheckbackslash{%
2969   \noexpand\def\noexpand\@gls@xdycheckbackslash##1\@backslashchar
2970     ##2\@backslashchar##3\noexpand\null{%
2971     \noexpand\@gls@tmpb=\noexpand\expandafter
2972       {\noexpand\@gls@checkedmkidx}%
2973     \noexpand\toks@={##1}%
2974     \noexpand\ifx\noexpand\null##2\noexpand\null
2975       \noexpand\ifx\noexpand\null##3\noexpand\null

```

```

2976 \noexpand\edef\noexpand\@gls@checkedmkidx{%
2977 \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@}%
2978 \noexpand\def\noexpand\@gls@xdycheckbackslash{\relax}%
2979 \noexpand\else
2980 \noexpand\edef\noexpand\@gls@checkedmkidx{%
2981 \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
2982 \@backslashchar\@backslashchar\@backslashchar\@backslashchar}%
2983 \noexpand\def\noexpand\@gls@xdycheckbackslash{%
2984 \noexpand\@gls@xdycheckbackslash##3\noexpand\null}%
2985 \noexpand\fi
2986 \noexpand\else
2987 \noexpand\edef\noexpand\@gls@checkedmkidx{%
2988 \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
2989 \@backslashchar\@backslashchar}%
2990 \noexpand\ifx\noexpand\null##3\noexpand\null
2991 \noexpand\def\noexpand\@gls@xdycheckbackslash{%
2992 \noexpand\@gls@xdycheckbackslash##2\@backslashchar
2993 \@backslashchar\noexpand\null}%
2994 \noexpand\else
2995 \noexpand\def\noexpand\@gls@xdycheckbackslash{%
2996 \noexpand\@gls@xdycheckbackslash##2\@backslashchar
2997 ##3\noexpand\null}%
2998 \noexpand\fi
2999 \noexpand\fi
3000 \noexpand\@gls@xdycheckbackslash
3001 }%
3002 }

```

Now go ahead and define \@gls@xdycheckbackslash

```

3003 \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.

```

3004 \ifcsundef{hyperlink}%
3005 {%
3006 \gdef\@glslink#1#2{#2}%
3007 }%
3008 {%
3009 \gdef\@glslink#1#2{\hyperlink{#1}{#2}}%
3010 }

```

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

```

3011 \newlength\gls@tmplen \ifcsundef{hypertarget}%
3012 {%
3013 \gdef\@glstarget#1#2{#2}%
3014 }%
3015 {%

```

```

3016 \gdef\@glstarget#1#2{%
3017     \settoheight{\gls@tmplen}{#2}%
3018     \raisebox{\gls@tmplen}{\hypertarget{#1}{}}#2%
3019 }%
3020 }

```

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

`\glsdisablehyper`

```

3021 \newcommand{\glsdisablehyper}{%
3022     \renewcommand*\@glslink[2]{\hyperlink{##1}{##2}}%
3023     \renewcommand*\@glstarget[2]{##2}%
3024 }

```

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

`\glsenablehyper`

```

3025 \newcommand{\glsenablehyper}{%
3026 \renewcommand*\@glslink[2]{\hyperlink{##1}{##2}}%
3027 \renewcommand*\@glstarget[2]{%
3028     \settoheight{\gls@tmplen}{##2}%
3029     \raisebox{\gls@tmplen}{\hypertarget{##1}{}}##2}}

```

Provide some convenience commands if not already defined:

```

3030 \providecommand{\@firstofthree}[3]{#1}
3031 \providecommand{\@secondofthree}[3]{#2}
3032 \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`

```

3033 \newrobustcmd*{\gls}{\@ifstar\@sgls\@gls}

```

Define the starred form:

\@sgls

```
3034 \newcommand*{\@sgls}[1] [] {\@gls[hyper=false,#1]}
```

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

\@gls

```
3035 \newcommand*{\@gls}[2] [] {%
3036   \new@ifnextchar[{\@gls@{#1}{#2}}{\@gls@{#1}{#2} []}]%
3037 }
```

\@gls@ Read in the final optional argument:

```
3038 \def\@gls@#1#2[#3]{%
3039   \glsdoifexists{#2}%
3040   {%
3041     \edef\@glo@type{\glsentrytype{#2}}%
3042     \let\glsifplural\@secondoftwo
3043     \let\glsifcaps\@firstofthree
3044     \let\glsifcustom\@empty
3045     \def\glsinsert{#3}%
```

Determine whether starred or unstarred version was used:

```
3046   \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper
3047   \setkeys{glslink}{hyper=true,#1}%
3048   \ifKV@glslink@hyper
3049     \let\glsifhyper\@firstoftwo
3050   \else
3051     \let\glsifhyper\@secondoftwo
3052   \fi
3053   \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper
```

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

```
3054   \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.

```
3055   \ifglsused{#2}%
3056   {%
3057     \@gls@link[#1]{#2}{\@glo@text}%
3058   }%
3059   {%
3060     \gls@checkisacronymlist\@glo@type
3061     \ifthenelse
3062       {\(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
3063        \OR \NOT\boolean{glsfirst}}
3064       {%
3065         \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
3066       }
```

```

3067     }%
3068     {%
3069     \@gls@link[#1]{#2}{\@glo@text}%
3070     }%
3071     }%

```

Indicate that this entry has now been used

```

3072     \ifKV@glslink@local
3073     \glslocalunset{#2}%
3074     \else
3075     \glsunset{#2}%
3076     \fi
3077     }%
3078 }

```

\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

```

3079 \newrobustcmd*{\Gls}{\@ifstar\@sGls\@Gls}

```

Define the starred form:

```

3080 \newcommand*{\@sGls}[1][\@Gls[hyper=false,#1]]{

```

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

```

3081 \newcommand*{\@Gls}[2][\@Gls[#1]{#2}]{
3082   \new@ifnextchar[\@Gls@{#1}{#2}]{\@Gls@{#1}{#2}[]}%
3083 }

```

\@Gls@ Read in the final optional argument:

```

3084 \def\@Gls@#1#2[#3]{%
3085   \glsdoifexists{#2}%
3086   {%
3087     \edef\@glo@type{\glstrytype{#2}}%
3088     \let\glsifplural\@secondoftwo
3089     \let\glscapscase\@secondofthree
3090     \let\glscustomtext\@empty
3091     \def\glsinsert{#3}%

```

Determine whether starred or unstarred version was used:

```

3092     \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper
3093     \setkeys{glslink}{hyper=true,#1}%
3094     \ifKV@glslink@hyper
3095     \let\glsifhyper\@firstoftwo
3096     \else
3097     \let\glsifhyper\@secondoftwo

```

```

3098 \fi
3099 \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper
    Determine what the link text should be (this is stored in \@glo@text)
3100 \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.
3101 \ifglsused{#2}%
3102 {%
3103 \@gls@link[#1]{#2}{\@glo@text}%
3104 }%
3105 {%
3106 \gls@checkisacronymlist\@glo@type
3107 \ifthenelse
3108 {%
3109 \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
3110 \OR \NOT\boolean{glshyperfirst}%
3111 }%
3112 {%
3113 \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
3114 }%
3115 {%
3116 \@gls@link[#1]{#2}{\@glo@text}%
3117 }%
3118 }%
    Indicate that this entry has now been used
3119 \ifKV@glslink@local
3120 \glslocalunset{#2}%
3121 \else
3122 \glsunset{#2}%
3123 \fi
3124 }%
3125 }

```

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

\GLS

```

3126 \newrobustcmd*{\GLS}{\@ifstar\@sGLS\@GLS}

```

Define the starred form:

```

3127 \newcommand*{\@sGLS}[1] [] {\@GLS[hyper=false,#1]}

```

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

```

3128 \newcommand*{\@GLS}[2] [] {%
3129 \new@ifnextchar[\@GLS@{#1}{#2}}{\@GLS@{#1}{#2} []}%
3130 }

```


\@GLS@ Read in the final optional argument:

```
3131 \def\@GLS@#1#2[#3]{%
3132   \glsdoifexists{#2}%
3133   {%
3134     \edef\@glo@type{\glsentrytype{#2}}%

3135     \let\glsifplural\@secondoftwo
3136     \let\glsifscapscase\@thirdofthree
3137     \let\glsifcustomtext\@empty
3138     \def\glsinsert{#3}%
```

Determine whether starred or unstarred version was used:

```
3139   \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper
3140   \setkeys{glslink}{hyper=true,#1}%
3141   \ifKV@glslink@hyper
3142     \let\glsifhyper\@firstoftwo
3143   \else
3144     \let\glsifhyper\@secondoftwo
3145   \fi
3146   \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper
```

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

```
3147   \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.

```
3148   \ifglsused{#2}%
3149   {%
3150     \@gls@link[#1]{#2}{\@glo@text}%
3151   }%
3152   {%
3153     \gls@checkisacronymlist\@glo@type
3154     \ifthenelse
3155     {%
3156       \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
3157       \OR \NOT\boolean{glshyperfirst}}{%
3158       \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
3159     }%
3160     {%
3161       \@gls@link[#1]{#2}{\@glo@text}%
3162     }%
3163   }%
```

Indicate that this entry has now been used

```
3164   \ifKV@glslink@local
3165   \glslocalunset{#2}%
3166 \else
3167   \glsunset{#2}%
3168 \fi
3169 }%
```

3170 }

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

`\glspl`

```
3171 \newrobustcmd*{\glspl}{\@ifstar\sglspl\glspl}
```

Define the starred form:

```
3172 \newcommand*{\sglspl}[1][\@glspl[hyper=false,#1]]
```

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

```
3173 \newcommand*{\glspl}[2][\%
```

```
3174 \new@ifnextchar[\@glspl@{#1}{#2}]{\@glspl@{#1}{#2}[]}%
```

```
3175 }
```

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

```
3176 \def\@glspl@#1#2[#3]{%
```

```
3177 \glsdoifexists{#2}%
```

```
3178 {%
```

```
3179 \edef\@glo@type{\glsentrytype{#2}}%
```

```
3180 \let\glsifplural\@firstoftwo
```

```
3181 \let\glsifscapscase\@firstofthree
```

```
3182 \let\glsifcustomtext\@empty
```

```
3183 \def\glsinsert{#3}%
```

Determine whether starred or unstarred version was used:

```
3184 \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper
```

```
3185 \setkeys{glslink}{hyper=true,#1}%
```

```
3186 \ifKV@glslink@hyper
```

```
3187 \let\glsifhyper\@firstoftwo
```

```
3188 \else
```

```
3189 \let\glsifhyper\@secondoftwo
```

```
3190 \fi
```

```
3191 \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper
```

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

```
3192 \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.

```
3193 \ifglsused{#2}%
```

```
3194 {%
```

```
3195 \@gls@link[#1]{#2}{\@glo@text}%
```

```
3196 }%
```

```
3197 {%
```

```
3198 \gls@checkisacronymlist\@glo@type
```

```
3199 \ifthenelse
```

```
3200 {%
```

```

3201      \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote})\
3202      \OR \NOT\boolean{glshyperfirst}%
3203  }%
3204  {%
3205      \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
3206  }%
3207  {%
3208      \@gls@link[#1]{#2}{\@glo@text}%
3209  }%
3210  }%

```

Indicate that this entry has now been used

```

3211      \ifKV@glslink@local
3212      \glslocalunset{#2}%
3213      \else
3214      \glsunset{#2}%
3215      \fi
3216  }%
3217 }

```

`\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`

```

3218 \newrobustcmd*{\Glspl}{\@ifstar\@sGlspl\@Glspl}

```

Define the starred form:

```

3219 \newcommand*{\@sGlspl}[1][\@Glspl[hyper=false,#1]]

```

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

```

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

```

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

```

3223 \def\@Glspl@#1#2[#3]{%
3224   \glsdoifexists{#2}%
3225   {%
3226     \edef\@glo@type{\glsentrytype{#2}}%
3227     \let\glsifplural\@firstoftwo
3228     \let\glscapscase\@secondofthree
3229     \let\glscustomtext\@empty
3230     \def\glsinsert{#3}%

```

Determine whether starred or unstarred version was used:

```

3231     \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper
3232     \setkeys{glslink}{hyper=true,#1}%

```

```

3233 \ifKV@glslink@hyper
3234 \let\glsifhyper\@firstoftwo
3235 \else
3236 \let\glsifhyper\@secondoftwo
3237 \fi
3238 \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper

```

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.

```

3239 \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.
3240 \ifglsused{#2}%
3241 {%
3242 \@gls@link[#1]{#2}{\@glo@text}%
3243 }%
3244 {%
3245 \gls@checkisacronymlist\@glo@type
3246 \ifthenelse
3247 {%
3248 \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
3249 \OR \NOT\boolean{glshyperfirst}%
3250 }%
3251 {%
3252 \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
3253 }%
3254 {%
3255 \@gls@link[#1]{#2}{\@glo@text}%
3256 }%
3257 }%

```

Indicate that this entry has now been used

```

3258 \ifKV@glslink@local
3259 \glslocalunset{#2}%
3260 \else
3261 \glsunset{#2}%
3262 \fi
3263 }%
3264 }

```

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

\GLSp1

```

3265 \newrobustcmd*{\GLSp1}{\@ifstar\@sGLSp1\@GLSp1}

```

Define the starred form:

```

3266 \newcommand*{\@sGLSp1}[1][\@GLSp1[hyper=false,#1]]{

```

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

```

3267 \newcommand*{\@GLSp1}[2][]{%
3268   \new@ifnextchar[{\@GLSp1@{#1}{#2}}{\@GLSp1@{#1}{#2}[]}%
3269 }

```

\@GLSp1 Read in the final optional argument:

```

3270 \def\@GLSp1@#1#2[#3]{%
3271   \glsdoifexists{#2}%
3272   {%
3273     \edef\@glo@type{\glsentrytype{#2}}%

3274     \let\glsifplural\@firstoftwo
3275     \let\glscapscase\@thirdofthree
3276     \let\glscustomtext\@empty
3277     \def\glsinsert{#3}%

```

Determine whether starred or unstarred version was used:

```

3278   \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper
3279   \setkeys{glslink}{hyper=true,#1}%
3280   \ifKV@glslink@hyper
3281     \let\glsifhyper\@firstoftwo
3282   \else
3283     \let\glsifhyper\@secondoftwo
3284   \fi
3285   \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper

```

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

```

3286   \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.

```

3287   \ifglsused{#2}%
3288   {%
3289     \@gls@link[#1]{#2}{\@glo@text}%
3290   }%
3291   {%
3292     \gls@checkisacronymlist\@glo@type
3293     \ifthenelse
3294     {%
3295       \(\boolean{glsisacronymlist}\AND \boolean{glsacrfootnote}\)
3296       \OR \NOT\boolean{glshyperfirst}%
3297     }%
3298     {%
3299       \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
3300     }%
3301     {%
3302       \@gls@link[#1]{#2}{\@glo@text}%
3303     }%

```

```

3304 }%
      Indicate that this entry has now been used
3305 \ifKV@glslink@local
3306 \glslocalunset{#2}%
3307 \else
3308 \glsunset{#2}%
3309 \fi
3310 }%
3311 }

```

`\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:

```

3312 \newrobustcmd*{\glsdisp}{\@ifstar\@sglsdisp\@glsdisp}

```

Define the starred form:

```

\@sgls
3313 \newcommand*{\@sglsdisp}[1][ ]{\@glsdisp[hyper=false,#1]}

```

Defined the un-starred form.

```

\@glsdisp
3314 \newcommand*{\@glsdisp}[3][ ]{%
3315 \glsdoifexists{#2}{%

3316 \edef\@glo@type{\glsentrytype{#2}}%

3317 \let\glsifplural\@secondoftwo
3318 \let\glsifscapscase\@firstofthree
3319 \def\glsifcustomtext{#3}%
3320 \def\glsifinsert{}}%

```

Determine whether starred or unstarred version was used:

```

3321 \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper
3322 \setkeys{glslink}{hyper=true,#1}%
3323 \ifKV@glslink@hyper
3324 \let\glsifhyper\@firstoftwo
3325 \else
3326 \let\glsifhyper\@secondoftwo
3327 \fi
3328 \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper

```

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

```

3329 \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.

```

3330 \ifglsused{#2}%
3331 {%
3332 \@gls@link[#1]{#2}{\@glo@text}%
3333 }%
3334 {%
3335 \gls@checkisacronymlist\@glo@type
3336 \ifthenelse{(\boolean{@glsisacronymlist}\AND
3337 \boolean{glsacrfootnote}) \OR \NOT\boolean{glshyperfirst}}%
3338 {%
3339 \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
3340 }%
3341 {%
3342 \@gls@link[#1]{#2}{\@glo@text}%
3343 }%
3344 }%

```

Indicate that this entry has now been used

```

3345 \ifKV@glslink@local
3346 \glslocalunset{#2}%
3347 \else
3348 \glsunset{#2}%
3349 \fi
3350 }%
3351 }

```

`\@gls@field@link`

```

3352 \newcommand{\@gls@field@link}[3]{%
3353 \glsdoifexists{#2}%
3354 {%
3355 \edef\@glo@type{\glsentrytype{#2}}%
3356 \@gls@link[#1]{#2}{#3}%
3357 }%
3358 }

```

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

`\glstext`

```

3359 \newrobustcmd*{\glstext}{\@ifstar\@sglstext\@glstext}

```

Define the starred form:

```

3360 \newcommand*{\@sglstext}[1][\@glstext[hyper=false,#1]]

```

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

```

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

```

Read in the final optional argument:

```

3363 \def\@glstext@#1#2[#3]{%

```

```

3364 \@gls@field@link{#1}{#2}{\glsentrytext{#2}#3}%
3365 }

```

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

\GLStext

```

3366 \newrobustcmd*{\GLStext}{\@ifstar\@sGLStext\@GLStext}

```

Define the starred form:

```

3367 \newcommand*{\@sGLStext}[1] [] {\@GLStext[hyper=false,#1]}

```

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

```

3368 \newcommand*{\@GLStext}[2] [] {%

```

```

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

```

Read in the final optional argument:

```

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

```

```

3371 \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentrytext{#2}#3}}%

```

```

3372 }

```

\Glsentrytext behaves like \glsentrytext except that the first letter of the text is converted to uppercase.

\Glsentrytext

```

3373 \newrobustcmd*{\Glsentrytext}{\@ifstar\@sGlsentrytext\@Glsentrytext}

```

Define the starred form:

```

3374 \newcommand*{\@sGlsentrytext}[1] [] {\@Glsentrytext[hyper=false,#1]}

```

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

```

3375 \newcommand*{\@Glsentrytext}[2] [] {%

```

```

3376 \new@ifnextchar[{\@Glsentrytext@{#1}{#2}}{\@Glsentrytext@{#1}{#2} []}]

```

Read in the final optional argument:

```

3377 \def\@Glsentrytext@#1#2[#3] {%

```

```

3378 \@gls@field@link{#1}{#2}{\Glsentrytext{#2}#3}%

```

```

3379 }

```

\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

```

3380 \newrobustcmd*{\glsfirst}{\@ifstar\@sglsfirst\@glsfirst}

```

Define the starred form:

```

3381 \newcommand*{\@sglsfirst}[1] [] {\@glsfirst[hyper=false,#1]}

```

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

```

3382 \newcommand*{\@glsfirst}[2] [] {%

```

```

3383 \new@ifnextchar[{\@glsfirst@{#1}{#2}}{\@glsfirst@{#1}{#2} []}]

```


Read in the final optional argument:

```
3384 \def\@glsfirst@#1#2[#3]{%
3385   \@gls@field@link{#1}{#2}{\glsentryfirst{#2}#3}%
3386 }
```

`\Glsfirst` behaves like `\glsfirst` except it displays the first letter in uppercase.

`\Glsfirst`

```
3387 \newrobustcmd*{\Glsfirst}{\@ifstar\@sGlsfirst\@Glsfirst}
```

Define the starred form:

```
3388 \newcommand*{\@sGlsfirst}[1][ ]{\@Glsfirst[hyper=false,#1]}
```

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

```
3389 \newcommand*{\@Glsfirst}[2][ ]{%
3390   \new@ifnextchar[{\@Glsfirst@{#1}{#2}}{\@Glsfirst@{#1}{#2}[ ]}}
```

Read in the final optional argument:

```
3391 \def\@Glsfirst@#1#2[#3]{%
3392   \@gls@field@link{#1}{#2}{\glsentryfirst{#2}#3}%
3393 }
```

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

`\GLSfirst`

```
3394 \newrobustcmd*{\GLSfirst}{\@ifstar\@sGLSfirst\@GLSfirst}
```

Define the starred form:

```
3395 \newcommand*{\@sGLSfirst}[1][ ]{\@GLSfirst[hyper=false,#1]}
```

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

```
3396 \newcommand*{\@GLSfirst}[2][ ]{%
3397   \new@ifnextchar[{\@GLSfirst@{#1}{#2}}{\@GLSfirst@{#1}{#2}[ ]}]
```

Read in the final optional argument:

```
3398 \def\@GLSfirst@#1#2[#3]{%
3399   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryfirst{#2}#3}}%
3400 }
```

`\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`

```
3401 \newrobustcmd*{\glsplural}{\@ifstar\@sglsplural\@glsplural}
```

Define the starred form:

```
3402 \newcommand*{\@sglsplural}[1][ ]{\@glsplural[hyper=false,#1]}
```

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

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

Read in the final optional argument:

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

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

\Glsplural

```
3408 \newrobustcmd*{\Glsplural}{\@ifstar\@sGlsplural\@Glsplural}
```

Define the starred form:

```
3409 \newcommand*{\@sGlsplural}[1] [] {\@Glsplural[hyper=false,#1]}
```

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

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

Read in the final optional argument:

```
3412 \def\@Glsplural@#1#2[#3] {%
3413   \@gls@field@link{#1}{#2}{\Glsentryplural{#2}#3}%
3414 }
```

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

\GLSplural

```
3415 \newrobustcmd*{\GLSplural}{\@ifstar\@sGLSplural\@GLSplural}
```

Define the starred form:

```
3416 \newcommand*{\@sGLSplural}[1] [] {\@GLSplural[hyper=false,#1]}
```

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

```
3417 \newcommand*{\@GLSplural}[2] [] {%
3418   \new@ifnextchar[{\@GLSplural@{#1}{#2}}{\@GLSplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3419 \def\@GLSplural@#1#2[#3] {%
3420   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryplural{#2}#3}}%
3421 }
```

\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

```
3422 \newrobustcmd*{\glsfirstplural}{\@ifstar\@sglsfirstplural\@glsfirstplural}
```

Define the starred form:

```
3423 \newcommand*{\@sglsfirstplural}[1] [] {\@glsfirstplural[hyper=false,#1]}
```

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

```
3424 \newcommand*{\@glsfirstplural}[2] [] {%
```

```
3425   \new@ifnextchar[{\@glsfirstplural@{#1}{#2}}{\@glsfirstplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3426 \def\@glsfirstplural@#1#2[#3] {%
```

```
3427   \@gls@field@link{#1}{#2}{\glsentryfirstplural{#2}#3}%
```

```
3428 }
```

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

`\Glsfirstplural`

```
3429 \newrobustcmd*{\Glsfirstplural}{\@ifstar\@sGlsfirstplural\@Glsfirstplural}
```

Define the starred form:

```
3430 \newcommand*{\@sGlsfirstplural}[1] [] {\@Glsfirstplural[hyper=false,#1]}
```

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

```
3431 \newcommand*{\@Glsfirstplural}[2] [] {%
```

```
3432   \new@ifnextchar[{\@Glsfirstplural@{#1}{#2}}{\@Glsfirstplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3433 \def\@Glsfirstplural@#1#2[#3] {%
```

```
3434   \@gls@field@link{#1}{#2}{\Glsentryfirstplural{#2}#3}%
```

```
3435 }
```

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

`\GLSfirstplural`

```
3436 \newrobustcmd*{\GLSfirstplural}{\@ifstar\@sGLSfirstplural\@GLSfirstplural}
```

Define the starred form:

```
3437 \newcommand*{\@sGLSfirstplural}[1] [] {\@GLSfirstplural[hyper=false,#1]}
```

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

```
3438 \newcommand*{\@GLSfirstplural}[2] [] {%
```

```
3439   \new@ifnextchar[{\@GLSfirstplural@{#1}{#2}}{\@GLSfirstplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3440 \def\@GLSfirstplural@#1#2[#3] {%
```

```
3441   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryfirstplural{#2}#3}}%
```

```
3442 }
```

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

`\glsname`

```
3443 \newrobustcmd*{\glsname}{\@ifstar\sglsname\@glsname}
```

Define the starred form:

```
3444 \newcommand*{\@sglsname}[1] [] {\@glsname[hyper=false,#1]}
```

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

```
3445 \newcommand*{\@glsname}[2] [] {%
```

```
3446   \new@ifnextchar[{\@glsname@{#1}{#2}}{\@glsname@{#1}{#2} []}]
```

Read in the final optional argument:

```
3447 \def\@glsname@#1#2[#3] {%
```

```
3448   \@gls@field@link{#1}{#2}{\glsentryname{#2}#3}%
```

```
3449 }
```

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

`\Glsname`

```
3450 \newrobustcmd*{\Glsname}{\@ifstar\@sGlsname\@Glsname}
```

Define the starred form:

```
3451 \newcommand*{\@sGlsname}[1] [] {\@Glsname[hyper=false,#1]}
```

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

```
3452 \newcommand*{\@Glsname}[2] [] {%
```

```
3453   \new@ifnextchar[{\@Glsname@{#1}{#2}}{\@Glsname@{#1}{#2} []}]
```

Read in the final optional argument:

```
3454 \def\@Glsname@#1#2[#3] {%
```

```
3455   \@gls@field@link{#1}{#2}{\Glsentryname{#2}#3}%
```

```
3456 }
```

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

`\GLSname`

```
3457 \newrobustcmd*{\GLSname}{\@ifstar\@sGLSname\@GLSname}
```

Define the starred form:

```
3458 \newcommand*{\@sGLSname}[1] [] {\@GLSname[hyper=false,#1]}
```

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

```
3459 \newcommand*{\@GLSname}[2] [] {%
```

```
3460   \new@ifnextchar[{\@GLSname@{#1}{#2}}{\@GLSname@{#1}{#2} []}]
```

Read in the final optional argument:

```
3461 \def\@GLSname@#1#2[#3] {%
```

```
3462   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryname{#2}#3}}%
```

```
3463 }
```

`\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`

```
3464 \newrobustcmd*{\glsdesc}{\@ifstar\sglsdesc\glsdesc}
```

Define the starred form:

```
3465 \newcommand*{\sglsdesc}[1][\@glsdesc[hyper=false,#1]]
```

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

```
3466 \newcommand*{\glsdesc}[2][\%
```

```
3467 \new@ifnextchar[\@glsdesc@{#1}{#2}]{\@glsdesc@{#1}{#2}[]}]
```

Read in the final optional argument:

```
3468 \def\@glsdesc@#1#2[#3]{\%
```

```
3469 \gls@field@link{#1}{#2}{\glsentrydesc{#2}#3}%
```

```
3470 }
```

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

`\Glsdesc`

```
3471 \newrobustcmd*{\Glsdesc}{\@ifstar\sglsdesc\Glsdesc}
```

Define the starred form:

```
3472 \newcommand*{\sglsdesc}[1][\@Glsdesc[hyper=false,#1]]
```

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

```
3473 \newcommand*{\Glsdesc}[2][\%
```

```
3474 \new@ifnextchar[\@Glsdesc@{#1}{#2}]{\@Glsdesc@{#1}{#2}[]}]
```

Read in the final optional argument:

```
3475 \def\@Glsdesc@#1#2[#3]{\%
```

```
3476 \gls@field@link{#1}{#2}{\Glsentrydesc{#2}#3}%
```

```
3477 }
```

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

`\GLSdesc`

```
3478 \newrobustcmd*{\GLSdesc}{\@ifstar\sglsdesc\GLSdesc}
```

Define the starred form:

```
3479 \newcommand*{\sglsdesc}[1][\@GLSdesc[hyper=false,#1]]
```

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

```
3480 \newcommand*{\GLSdesc}[2][\%
```

```
3481 \new@ifnextchar[\@GLSdesc@{#1}{#2}]{\@GLSdesc@{#1}{#2}[]}]
```

Read in the final optional argument:

```
3482 \def\@GLSdesc@#1#2[#3]{%
3483   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentrydesc{#2}#3}}%
3484 }
```

`\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`

```
3485 \newrobustcmd*{\glsdescplural}{\@ifstar\sglsdescplural\@glsdescplural}
```

Define the starred form:

```
3486 \newcommand*{\@sglsdescplural}[1][\@glsdescplural[hyper=false,#1]]
```

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

```
3487 \newcommand*{\@glsdescplural}[2][\@%
3488   \new@ifnextchar[\@glsdescplural@{#1}{#2}]{\@glsdescplural@{#1}{#2}[]}]
```

Read in the final optional argument:

```
3489 \def\@glsdescplural@#1#2[#3]{%
3490   \@gls@field@link{#1}{#2}{\glsentrydescplural{#2}#3}%
3491 }
```

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

`\Glsdescplural`

```
3492 \newrobustcmd*{\Glsdescplural}{\@ifstar\@sGlsdescplural\@Glsdescplural}
```

Define the starred form:

```
3493 \newcommand*{\@sGlsdescplural}[1][\@Glsdescplural[hyper=false,#1]]
```

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

```
3494 \newcommand*{\@Glsdescplural}[2][\@%
3495   \new@ifnextchar[\@Glsdescplural@{#1}{#2}]{\@Glsdescplural@{#1}{#2}[]}]
```

Read in the final optional argument:

```
3496 \def\@Glsdescplural@#1#2[#3]{%
3497   \@gls@field@link{#1}{#2}{\Glsentrydescplural{#2}#3}%
3498 }
```

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

`\GLSdescplural`

```
3499 \newrobustcmd*{\GLSdescplural}{\@ifstar\@sGLSdescplural\@GLSdescplural}
```

Define the starred form:

```
3500 \newcommand*{\@sGLSdescplural}[1][\@GLSdescplural[hyper=false,#1]]
```

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

```
3501 \newcommand*{\@GLSdescplural}[2] [] {%
3502   \new@ifnextchar[{\@GLSdescplural@{#1}{#2}}{\@GLSdescplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3503 \def\@GLSdescplural@#1#2[#3] {%
3504   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentrydescplural{#2}#3}}%
3505 }
```

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

\glsymbol

```
3506 \newrobustcmd*{\glsymbol}{\@ifstar\@sglsymbol\@glsymbol}
```

Define the starred form:

```
3507 \newcommand*{\@sglsymbol}[1] [] {\@glsymbol[hyper=false,#1]}
```

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

```
3508 \newcommand*{\@glsymbol}[2] [] {%
3509   \new@ifnextchar[{\@glsymbol@{#1}{#2}}{\@glsymbol@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3510 \def\@glsymbol@#1#2[#3] {%
3511   \@gls@field@link{#1}{#2}{\glsentrysymbol{#2}#3}%
3512 }
```

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

\Glsymbol

```
3513 \newrobustcmd*{\Glsymbol}{\@ifstar\@sGlsymbol\@Glsymbol}
```

Define the starred form:

```
3514 \newcommand*{\@sGlsymbol}[1] [] {\@Glsymbol[hyper=false,#1]}
```

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

```
3515 \newcommand*{\@Glsymbol}[2] [] {%
3516   \new@ifnextchar[{\@Glsymbol@{#1}{#2}}{\@Glsymbol@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3517 \def\@Glsymbol@#1#2[#3] {%
3518   \@gls@field@link{#1}{#2}{\Glsentrysymbol{#2}#3}%
3519 }
```

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

\GLSsymbol

```
3520 \newrobustcmd*{\GLSsymbol}{\@ifstar\@sGLSsymbol\@GLSsymbol}
```

Define the starred form:

```
3521 \newcommand*{\@sGLSymbol}[1] [] {\@GLSymbol[hyper=false,#1]}
```

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

```
3522 \newcommand*{\@GLSymbol}[2] [] {%
```

```
3523   \new@ifnextchar[{\@GLSymbol@{#1}{#2}}{\@GLSymbol@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3524 \def\@GLSymbol@#1#2[#3] {%
```

```
3525   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentrysymbol{#2}#3}}%
```

```
3526 }
```

`\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`

```
3527 \newrobustcmd*{\glsymbolplural}{\@ifstar\@sglsymbolplural\@glsymbolplural}
```

Define the starred form:

```
3528 \newcommand*{\@sglsymbolplural}[1] [] {\@glsymbolplural[hyper=false,#1]}
```

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

```
3529 \newcommand*{\@glsymbolplural}[2] [] {%
```

```
3530   \new@ifnextchar[{\@glsymbolplural@{#1}{#2}}{\@glsymbolplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3531 \def\@glsymbolplural@#1#2[#3] {%
```

```
3532   \@gls@field@link{#1}{#2}{\glsentrysymbolplural{#2}#3}}%
```

```
3533 }
```

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

`\Glsymbolplural`

```
3534 \newrobustcmd*{\Glsymbolplural}{\@ifstar\@sGlsymbolplural\@Glsymbolplural}
```

Define the starred form:

```
3535 \newcommand*{\@sGlsymbolplural}[1] [] {\@Glsymbolplural[hyper=false,#1]}
```

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

```
3536 \newcommand*{\@Glsymbolplural}[2] [] {%
```

```
3537   \new@ifnextchar[{\@Glsymbolplural@{#1}{#2}}{\@Glsymbolplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3538 \def\@Glsymbolplural@#1#2[#3] {%
```

```
3539   \@gls@field@link{#1}{#2}{\Glsentrysymbolplural{#2}#3}}%
```

```
3540 }
```

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

\GLSsymbolplural

```
3541 \newrobustcmd*{\GLSsymbolplural}{\@ifstar\@sGLSsymbolplural\@GLSsymbolplural}
```

Define the starred form:

```
3542 \newcommand*{\@sGLSsymbolplural}[1] [] {\@GLSsymbolplural[hyper=false,#1]}
```

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

```
3543 \newcommand*{\@GLSsymbolplural}[2] [] {%
```

```
3544   \new@ifnextchar[{\@GLSsymbolplural@{#1}{#2}}{\@GLSsymbolplural@{#1}{#2} []}]
```

Read in the final optional argument:

```
3545 \def\@GLSsymbolplural@#1#2[#3] {%
```

```
3546   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentrysymbolplural{#2}#3}}%
```

```
3547 }
```

\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

```
3548 \newrobustcmd*{\glsuseri}{\@ifstar\@sglsuseri\@glsuseri}
```

Define the starred form:

```
3549 \newcommand*{\@sglsuseri}[1] [] {\@glsuseri[hyper=false,#1]}
```

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

```
3550 \newcommand*{\@glsuseri}[2] [] {%
```

```
3551   \new@ifnextchar[{\@glsuseri@{#1}{#2}}{\@glsuseri@{#1}{#2} []}]
```

Read in the final optional argument:

```
3552 \def\@glsuseri@#1#2[#3] {%
```

```
3553   \@gls@field@link{#1}{#2}{\glsentryuseri{#2}#3}%
```

```
3554 }
```

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

\Glsuseri

```
3555 \newrobustcmd*{\Glsuseri}{\@ifstar\@sGlsuseri\@Glsuseri}
```

Define the starred form:

```
3556 \newcommand*{\@sGlsuseri}[1] [] {\@Glsuseri[hyper=false,#1]}
```

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

```
3557 \newcommand*{\@Glsuseri}[2] [] {%
```

```
3558   \new@ifnextchar[{\@Glsuseri@{#1}{#2}}{\@Glsuseri@{#1}{#2} []}]
```

Read in the final optional argument:

```
3559 \def\@Glsuseri@#1#2[#3] {%
```

```
3560   \@gls@field@link{#1}{#2}{\Glsentryuseri{#2}#3}%
```

```
3561 }
```

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

`\GLSuseri`

```
3562 \newrobustcmd*{\GLSuseri}{\@ifstar\@sGLSuseri\@GLSuseri}
```

Define the starred form:

```
3563 \newcommand*{\@sGLSuseri}[1] [] {\@GLSuseri[hyper=false,#1]}
```

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

```
3564 \newcommand*{\@GLSuseri}[2] [] {%
```

```
3565   \new@ifnextchar[{\@GLSuseri@{#1}{#2}}{\@GLSuseri@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3566 \def\@GLSuseri@#1#2[#3] {%
```

```
3567   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryuseri{#2}#3}}%
```

```
3568 }
```

`\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`

```
3569 \newrobustcmd*{\glsuserii}{\@ifstar\@sglsuserii\@glsuserii}
```

Define the starred form:

```
3570 \newcommand*{\@sglsuserii}[1] [] {\@glsuserii[hyper=false,#1]}
```

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

```
3571 \newcommand*{\@glsuserii}[2] [] {%
```

```
3572   \new@ifnextchar[{\@glsuserii@{#1}{#2}}{\@glsuserii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3573 \def\@glsuserii@#1#2[#3] {%
```

```
3574   \@gls@field@link{#1}{#2}{\glsentryuserii{#2}#3}}%
```

```
3575 }
```

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

`\Glsuserii`

```
3576 \newrobustcmd*{\Glsuserii}{\@ifstar\@sGlsuserii\@Glsuserii}
```

Define the starred form:

```
3577 \newcommand*{\@sGlsuserii}[1] [] {\@Glsuserii[hyper=false,#1]}
```

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

```
3578 \newcommand*{\@Glsuserii}[2] [] {%
```

```
3579   \new@ifnextchar[{\@Glsuserii@{#1}{#2}}{\@Glsuserii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3580 \def\@Glsuserii@#1#2[#3]{%
3581   \@gls@field@link{#1}{#2}{\Glsentryuserii{#2}#3}%
3582 }
```

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

\Glsuserii

```
3583 \newrobustcmd*{\Glsuserii}{\@ifstar\@sGlsuserii\@Glsuserii}
```

Define the starred form:

```
3584 \newcommand*{\@sGlsuserii}[1][\@Glsuserii[hyper=false,#1]]
```

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

```
3585 \newcommand*{\@Glsuserii}[2][\@Glsuserii[hyper=false,#1]]{
3586   \new@ifnextchar[\@Glsuserii@{#1}{#2}}{\@Glsuserii@{#1}{#2}}}
```

Read in the final optional argument:

```
3587 \def\@Glsuserii@#1#2[#3]{%
3588   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\Glsentryuserii{#2}#3}}%
3589 }
```

\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

```
3590 \newrobustcmd*{\glsuseriii}{\@ifstar\@sglsuseriii\@glsuseriii}
```

Define the starred form:

```
3591 \newcommand*{\@sglsuseriii}[1][\@glsuseriii[hyper=false,#1]]
```

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

```
3592 \newcommand*{\@glsuseriii}[2][\@glsuseriii[hyper=false,#1]]{
3593   \new@ifnextchar[\@glsuseriii@{#1}{#2}}{\@glsuseriii@{#1}{#2}}}
```

Read in the final optional argument:

```
3594 \def\@glsuseriii@#1#2[#3]{%
3595   \@gls@field@link{#1}{#2}{\Glsentryuseriii{#2}#3}%
3596 }
```

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

\Glsuseriii

```
3597 \newrobustcmd*{\Glsuseriii}{\@ifstar\@sGlsuseriii\@Glsuseriii}
```

Define the starred form:

```
3598 \newcommand*{\@sGlsuseriii}[1][\@Glsuseriii[hyper=false,#1]]
```

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

```
3599 \newcommand*{\@Glsuseriii}[2] [] {%
3600   \new@ifnextchar[{\@Glsuseriii@{#1}{#2}}{\@Glsuseriii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3601 \def\@Glsuseriii@#1#2[#3] {%
3602   \@gls@field@link{#1}{#2}{\Glsentryuseriii{#2}#3}%
3603 }
```

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

\Glsuseriii

```
3604 \newrobustcmd*{\Glsuseriii}{\@ifstar\@sGlsuseriii\@Glsuseriii}
```

Define the starred form:

```
3605 \newcommand*{\@sGlsuseriii}[1] [] {\@Glsuseriii[hyper=false,#1]}
```

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

```
3606 \newcommand*{\@Glsuseriii}[2] [] {%
3607   \new@ifnextchar[{\@Glsuseriii@{#1}{#2}}{\@Glsuseriii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3608 \def\@Glsuseriii@#1#2[#3] {%
3609   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\Glsentryuseriii{#2}#3}}%
3610 }
```

\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

```
3611 \newrobustcmd*{\glsuseriv}{\@ifstar\@sglsuseriv\@glsuseriv}
```

Define the starred form:

```
3612 \newcommand*{\@sglsuseriv}[1] [] {\@glsuseriv[hyper=false,#1]}
```

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

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

Read in the final optional argument:

```
3615 \def\@glsuseriv@#1#2[#3] {%
3616   \@gls@field@link{#1}{#2}{\Glsentryuseriv{#2}#3}%
3617 }
```

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

\Glsuseriv

```
3618 \newrobustcmd*{\Glsuseriv}{\@ifstar\@sGlsuseriv\@Glsuseriv}
```

Define the starred form:

```
3619 \newcommand*{\@sGlsuseriv}[1] [] {\@Glsuseriv[hyper=false,#1]}
```

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

```
3620 \newcommand*{\@Glsuseriv}[2] [] {%
```

```
3621   \new@ifnextchar[{\@Glsuseriv@{#1}{#2}}{\@Glsuseriv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3622 \def\@Glsuseriv@#1#2[#3] {%
```

```
3623   \@gls@field@link{#1}{#2}{\Glsentryuseriv{#2}#3}%
```

```
3624 }
```

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

\Glsuseriv

```
3625 \newrobustcmd*{\Glsuseriv}{\@ifstar\@sGlsuseriv\@Glsuseriv}
```

Define the starred form:

```
3626 \newcommand*{\@sGlsuseriv}[1] [] {\@Glsuseriv[hyper=false,#1]}
```

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

```
3627 \newcommand*{\@Glsuseriv}[2] [] {%
```

```
3628   \new@ifnextchar[{\@Glsuseriv@{#1}{#2}}{\@Glsuseriv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3629 \def\@Glsuseriv@#1#2[#3] {%
```

```
3630   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\Glsentryuseriv{#2}#3}}%
```

```
3631 }
```

\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

```
3632 \newrobustcmd*{\glsuserv}{\@ifstar\@sglsuserv\@glsuserv}
```

Define the starred form:

```
3633 \newcommand*{\@sglsuserv}[1] [] {\@glsuserv[hyper=false,#1]}
```

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

```
3634 \newcommand*{\@glsuserv}[2] [] {%
```

```
3635   \new@ifnextchar[{\@glsuserv@{#1}{#2}}{\@glsuserv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3636 \def\@glsuserv@#1#2[#3] {%
```

```
3637   \@gls@field@link{#1}{#2}{\Glsentryuserv{#2}#3}%
```

```
3638 }
```

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

`\Glsuserv`

```
3639 \newrobustcmd*{\Glsuserv}{\@ifstar\@sGlsuserv\@Glsuserv}
```

Define the starred form:

```
3640 \newcommand*{\@sGlsuserv}[1] [] {\@Glsuserv[hyper=false,#1]}
```

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

```
3641 \newcommand*{\@Glsuserv}[2] [] {%
```

```
3642 \new@ifnextchar [{\@Glsuserv@{#1}{#2}}{\@Glsuserv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3643 \def\@Glsuserv@#1#2[#3] {%
```

```
3644 \@gls@field@link{#1}{#2}{\Glsentryuserv{#2}#3}%
```

```
3645 }
```

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

`\GLSuserv`

```
3646 \newrobustcmd*{\GLSuserv}{\@ifstar\@sGLSuserv\@GLSuserv}
```

Define the starred form:

```
3647 \newcommand*{\@sGLSuserv}[1] [] {\@GLSuserv[hyper=false,#1]}
```

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

```
3648 \newcommand*{\@GLSuserv}[2] [] {%
```

```
3649 \new@ifnextchar [{\@GLSuserv@{#1}{#2}}{\@GLSuserv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3650 \def\@GLSuserv@#1#2[#3] {%
```

```
3651 \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryuserv{#2}#3}}%
```

```
3652 }
```

`\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`

```
3653 \newrobustcmd*{\glsuservi}{\@ifstar\@sglsuservi\@glsuservi}
```

Define the starred form:

```
3654 \newcommand*{\@sglsuservi}[1] [] {\@glsuservi[hyper=false,#1]}
```

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

```
3655 \newcommand*{\@glsuservi}[2] [] {%
```

```
3656 \new@ifnextchar [{\@glsuservi@{#1}{#2}}{\@glsuservi@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3657 \def\@glsuservi@#1#2[#3] {%
```

```
3658 \@gls@field@link{#1}{#2}{\glsentryuservi{#2}#3}%
```

```
3659 }
```

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

\Glsuservi

```
3660 \newrobustcmd*{\Glsuservi}{\@ifstar\sGlsuservi\@Glsuservi}
```

Define the starred form:

```
3661 \newcommand*{\sGlsuservi}[1] [] {\@Glsuservi[hyper=false,#1]}
```

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

```
3662 \newcommand*{\@Glsuservi}[2] [] {%
```

```
3663   \new@ifnextchar[{\@Glsuservi@{#1}{#2}}{\@Glsuservi@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3664 \def\@Glsuservi@#1#2[#3] {%
```

```
3665   \@gls@field@link{#1}{#2}{\Glsentryuservi{#2}#3}%
```

```
3666 }
```

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

\GLSuservi

```
3667 \newrobustcmd*{\GLSuservi}{\@ifstar\sGLSuservi\@GLSuservi}
```

Define the starred form:

```
3668 \newcommand*{\sGLSuservi}[1] [] {\@GLSuservi[hyper=false,#1]}
```

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

```
3669 \newcommand*{\@GLSuservi}[2] [] {%
```

```
3670   \new@ifnextchar[{\@GLSuservi@{#1}{#2}}{\@GLSuservi@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3671 \def\@GLSuservi@#1#2[#3] {%
```

```
3672   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryuservi{#2}#3}}%
```

```
3673 }
```

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

\acrshort

```
3674 \newrobustcmd*{\acrshort}{\@ifstar\s@acrshort\@ns@acrshort}
```

Define the starred form:

```
3675 \newcommand*{\s@acrshort}[2] [] {%
```

```
3676   \new@ifnextchar[{\@acrshort{hyper=false,#1}{#2}}%
```

```
3677   {\@acrshort{hyper=false,#1}{#2} []}]%
```

```
3678 }
```

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

```
3679 \newcommand*{\ns@acrshort}[2] [] {%
```

```
3680   \new@ifnextchar[{\@acrshort{#1}{#2}}{\@acrshort{#1}{#2} []}]%
```

```
3681 }
```

Read in the final optional argument:

```

3682 \def\@acrshort#1#2[#3]{%
3683   \glsdoifexists{#2}%
3684   {%
3685     \edef\@glo@type{\glsentrytype{#2}}%

3686     \let\glsifplural\@secondoftwo
3687     \let\glscapscase\@firstofthree
3688     \let\glsinsert\@empty
3689     \def\glscustomtext{%
3690       \acronymfont{\glsentryshort{#2}}#3%
3691     }%

```

Determine whether starred or unstarred version was used:

```

3692   \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper
3693   \setkeys{glslink}{hyper=true,#1}%
3694   \ifKV@glslink@hyper
3695     \let\glsifhyper\@firstoftwo
3696   \else
3697     \let\glsifhyper\@secondoftwo
3698   \fi
3699   \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper

```

Call \@gls@link

```

3700   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3701 }%
3702 }

```

\Acrshort

```

3703 \newrobustcmd*{\Acrshort}{\@ifstar\s@Acrshort\ns@Acrshort}

```

Define the starred form:

```

3704 \newcommand*{\s@Acrshort}[2][]{%
3705   \new@ifnextchar[{\@Acrshort{hyper=false,#1}{#2}}%
3706   {\@Acrshort{hyper=false,#1}{#2}[]}%
3707 }

```

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

```

3708 \newcommand*{\ns@Acrshort}[2][]{%
3709   \new@ifnextchar[{\@Acrshort{#1}{#2}}{\@Acrshort{#1}{#2}[]}%
3710 }

```

Read in the final optional argument:

```

3711 \def\@Acrshort#1#2[#3]{%
3712   \glsdoifexists{#2}%
3713   {%
3714     \edef\@glo@type{\glsentrytype{#2}}%

```



```

3715 \def\glslabel{#2}%
3716 \let\glsifplural\@secondoftwo
3717 \let\glscapscase\@secondofthree
3718 \let\glsinsert\@empty
3719 \def\glscustomtext{%
3720 \acronymfont{\Glsentryshort{#2}}#3%
3721 }%

```

Determine whether starred or unstarred version was used:

```

3722 \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper
3723 \setkeys{glslink}{hyper=true,#1}%
3724 \ifKV@glslink@hyper
3725 \let\glsifhyper\@firstoftwo
3726 \else
3727 \let\glsifhyper\@secondoftwo
3728 \fi
3729 \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper

```

Call \@gls@link

```

3730 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3731 }%
3732 }

```

\ACRshort

```

3733 \newrobustcmd*{\ACRshort}{\@ifstar\s@ACRshort\ns@ACRshort}

```

Define the starred form:

```

3734 \newcommand*{\s@ACRshort}[2][]{%
3735 \new@ifnextchar[{\@ACRshort{hyper=false,#1}{#2}}%
3736 {\@ACRshort{hyper=false,#1}{#2}[]}%
3737 }

```

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

```

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

```

Read in the final optional argument:

```

3741 \def\@ACRshort#1#2[#3]{%
3742 \glsdoifexists{#2}%
3743 {%
3744 \edef\@glo@type{\glsentrytype{#2}}%

3745 \def\glslabel{#2}%
3746 \let\glsifplural\@secondoftwo
3747 \let\glscapscase\@thirdofthree
3748 \let\glsinsert\@empty
3749 \def\glscustomtext{%
3750 \mfirstucMakeUppercase{\acronymfont{\glsentryshort{#2}}#3}%
3751 }%

```

Determine whether starred or unstarred version was used:

```

3752 \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper
3753 \setkeys{glslink}{hyper=true,#1}%
3754 \ifKV@glslink@hyper
3755 \let\glsifhyper\@firstoftwo
3756 \else
3757 \let\glsifhyper\@secondoftwo
3758 \fi
3759 \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper

```

Call \@gls@link

```

3760 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3761 }%
3762 }

```

Short plural:

\acrshortpl

```

3763 \newrobustcmd*{\acrshortpl}{\@ifstar\s@acrshortpl\@ns@acrshortpl}

```

Define the starred form:

```

3764 \newcommand*{\s@acrshortpl}[2] [] {%
3765 \new@ifnextchar[{\@acrshortpl{hyper=false,#1}{#2}}%
3766 {\@acrshortpl{hyper=false,#1}{#2} []}%
3767 }

```

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

```

3768 \newcommand*{\ns@acrshortpl}[2] [] {%
3769 \new@ifnextchar[{\@acrshortpl{#1}{#2}}{\@acrshortpl{#1}{#2} []}%
3770 }

```

Read in the final optional argument:

```

3771 \def\@acrshortpl#1#2[#3] {%
3772 \glsdoifexists{#2}%
3773 {%
3774 \edef\@glo@type{\glsentrytype{#2}}%
3775 \def\glslabel{#2}%
3776 \let\glsifplural\@firstoftwo
3777 \let\glscapscase\@firstofthree
3778 \let\glsinsert\@empty
3779 \def\glscustomtext{%
3780 \acronymfont{\glsentryshortpl{#2}}#3%
3781 }%

```

Determine whether starred or unstarred version was used:

```

3782 \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper
3783 \setkeys{glslink}{hyper=true,#1}%
3784 \ifKV@glslink@hyper
3785 \let\glsifhyper\@firstoftwo

```

```

3786 \else
3787 \let\glsifhyper\@secondoftwo
3788 \fi
3789 \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper
    Call \@gls@link
3790 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3791 }%
3792 }

```

\Acrshortpl

```

3793 \newrobustcmd*{\Acrshortpl}{\@ifstar\s@Acrshortpl\@ns@Acrshortpl}

```

Define the starred form:

```

3794 \newcommand*{\s@Acrshortpl}[2] [] {%
3795 \new@ifnextchar[{\@Acrshortpl{hyper=false,#1}{#2}}%
3796 {\@Acrshortpl{hyper=false,#1}{#2} []}%
3797 }

```

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

```

3798 \newcommand*{\ns@Acrshortpl}[2] [] {%
3799 \new@ifnextchar[{\@Acrshortpl{#1}{#2}}{\@Acrshortpl{#1}{#2} []}%
3800 }

```

Read in the final optional argument:

```

3801 \def\@Acrshortpl#1#2[#3] {%
3802 \glsdoifexists{#2}%
3803 {%
3804 \edef\@glo@type{\glentrytype{#2}}%
3805 \def\glslabel{#2}%
3806 \let\glsifplural\@firstoftwo
3807 \let\glscapscase\@secondofthree
3808 \let\glsinsert\@empty
3809 \def\glscustomtext{%
3810 \acronymfont{\Glsentryshortpl{#2}}#3%
3811 }%

```

Determine whether starred or unstarred version was used:

```

3812 \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper
3813 \setkeys{glslink}{hyper=true,#1}%
3814 \ifKV@glslink@hyper
3815 \let\glsifhyper\@firstoftwo
3816 \else
3817 \let\glsifhyper\@secondoftwo
3818 \fi
3819 \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper
    Call \@gls@link
3820 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%

```

```

3821 }%
3822 }

```

\ACRshortpl

```

3823 \newrobustcmd*{\ACRshortpl}{\@ifstar\s@ACRshortpl\ns@ACRshortpl}

```

Define the starred form:

```

3824 \newcommand*{\s@ACRshortpl}[2][]{%
3825   \new@ifnextchar[{\@ACRshortpl{hyper=false,#1}{#2}}{%
3826     {\@ACRshortpl{hyper=false,#1}{#2}[]}%
3827 }

```

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

```

3828 \newcommand*{\ns@ACRshortpl}[2][]{%
3829   \new@ifnextchar[{\@ACRshortpl{#1}{#2}}{\@ACRshortpl{#1}{#2}[]}%
3830 }

```

Read in the final optional argument:

```

3831 \def\@ACRshortpl#1#2[#3]{%
3832   \glsdoifexists{#2}%
3833   {%
3834     \edef\@glo@type{\glsentrytype{#2}}%

3835     \def\glslabel{#2}%
3836     \let\glsifplural\@firstoftwo
3837     \let\glscapscase\@thirdofthree
3838     \let\glsinsert\@empty
3839     \def\glscustomtext{%
3840       \mfirstucMakeUppercase{\acronymfont{\glsentryshortpl{#2}}#3}%
3841     }%

```

Determine whether starred or unstarred version was used:

```

3842   \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper
3843   \setkeys{glslink}{hyper=true,#1}%
3844   \ifKV@glslink@hyper
3845     \let\glsifhyper\@firstoftwo
3846   \else
3847     \let\glsifhyper\@secondoftwo
3848   \fi
3849   \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper

```

Call \@gls@link

```

3850   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3851 }%
3852 }

```

\acrlong

```

3853 \newrobustcmd*{\acrlong}{\@ifstar\s@acrlong\ns@acrlong}

```

Define the starred form:

```
3854 \newcommand*{\s@acrlong}[2] [] {%
3855   \new@ifnextchar[{\@acrlong{hyper=false,#1}{#2}}}%
3856   {\@acrlong{hyper=false,#1}{#2} []}%
3857 }
```

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

```
3858 \newcommand*{\ns@acrlong}[2] [] {%
3859   \new@ifnextchar[{\@acrlong{#1}{#2}}{\@acrlong{#1}{#2} []}%
3860 }
```

Read in the final optional argument:

```
3861 \def\@acrlong#1#2[#3] {%
3862   \glsdoifexists{#2}%
3863   {%
3864     \edef\@glo@type{\glsentrytype{#2}}%
3865     \def\glslabel{#2}%
3866     \let\glsifplural\@secondoftwo
3867     \let\glscapscase\@firstofthree
3868     \let\glsinsert\@empty
```

Determine whether starred or unstarred version was used:

```
3869     \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper
3870     \setkeys{glslink}{hyper=true,#1}%
3871     \ifKV@glslink@hyper
3872       \let\glsifhyper\@firstoftwo
3873     \else
3874       \let\glsifhyper\@secondoftwo
3875     \fi
3876     \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper
```

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

```
3877     \def\glscustomtext{%
3878       \glsentrylong{#2}#3%
3879     }%
```

Call \@gls@link

```
3880     \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3881   }%
3882 }
```

\Acrlong

```
3883 \newrobustcmd*{\Acrlong}{\@ifstar\s@Acrlong\ns@Acrlong}
```

Define the starred form:

```
3884 \newcommand*{\s@Acrlong}[2] [] {%
3885   \new@ifnextchar[{\@Acrlong{hyper=false,#1}{#2}}}%
3886   {\@Acrlong{hyper=false,#1}{#2} []}%
3887 }
```

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

```
3888 \newcommand*{\ns@Acrlong}[2] [] {%
3889   \new@ifnextchar[{\@Acrlong{#1}{#2}}{\@Acrlong{#1}{#2} []}%
3890 }
```

Read in the final optional argument:

```
3891 \def\@Acrlong#1#2[#3] {%
3892   \glsdoifexists{#2}%
3893   {%
3894     \edef\@glo@type{\glsentrytype{#2}}%
3895   }
3896   \def\glslabel{#2}%
3897   \let\glsifplural\@secondoftwo
3898   \let\glsifscapcase\@secondofthree
3899   \let\glsinsert\@empty
```

Determine whether starred or unstarred version was used:

```
3899   \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper
3900   \setkeys{glslink}{hyper=true,#1}%
3901   \ifKV@glslink@hyper
3902     \let\glsifhyper\@firstoftwo
3903   \else
3904     \let\glsifhyper\@secondoftwo
3905   \fi
3906   \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper
```

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

```
3907   \def\glscustomtext{%
3908     \Glsentrylong{#2}#3%
3909   }%
```

Call \@gls@link

```
3910   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3911 }%
3912 }
```

\ACRlong

```
3913 \newrobustcmd*{\ACRlong}{\@ifstar\s@ACRlong\ns@ACRlong}
```

Define the starred form:

```
3914 \newcommand*{\s@ACRlong}[2] [] {%
3915   \new@ifnextchar[{\@ACRlong{hyper=false,#1}{#2}}%
3916   {\@ACRlong{hyper=false,#1}{#2} []}%
3917 }
```

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

```
3918 \newcommand*{\ns@ACRlong}[2] [] {%
3919   \new@ifnextchar[{\@ACRlong{#1}{#2}}{\@ACRlong{#1}{#2} []}%
3920 }
```

Read in the final optional argument:

```

3921 \def\@ACRlong#1#2[#3]{%
3922   \glsdoifexists{#2}%
3923   {%
3924     \edef\@glo@type{\glsentrytype{#2}}%

3925   \def\glslabel{#2}%
3926   \let\glsifplural\@secondoftwo
3927   \let\glscapscase\@thirdofthree
3928   \let\glsinsert\@empty

```

Determine whether starred or unstarred version was used:

```

3929   \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper
3930   \setkeys{glslink}{hyper=true,#1}%
3931   \ifKV@glslink@hyper
3932     \let\glsifhyper\@firstoftwo
3933   \else
3934     \let\glsifhyper\@secondoftwo
3935   \fi
3936   \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper

```

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

```

3937   \def\glscustomtext{%
3938     \mfirstucMakeUppercase{\glsentrylong{#2}#3}%
3939   }%

```

Call \@gls@link

```

3940   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3941 }%
3942 }

```

Short plural:

\acrlongpl

```

3943 \newrobustcmd*{\acrlongpl}{\@ifstar\s@acrlongpl\ns@acrlongpl}

```

Define the starred form:

```

3944 \newcommand*{\s@acrlongpl}[2][ ]{%
3945   \new@ifnextchar[{\@acrlongpl{hyper=false,#1}{#2}}%
3946     {\@acrlongpl{hyper=false,#1}{#2}[ ]}%
3947 }

```

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

```

3948 \newcommand*{\ns@acrlongpl}[2][ ]{%
3949   \new@ifnextchar[{\@acrlongpl{#1}{#2}}{\@acrlongpl{#1}{#2}[ ]}%
3950 }

```

Read in the final optional argument:

```

3951 \def\@acrlongpl#1#2[#3]{%
3952   \glsdoifexists{#2}%

```

```

3953  {%
3954    \edef\@glo@type{\glsentrytype{#2}}%

3955    \def\glslabel{#2}%
3956    \let\glsifplural\@firstoftwo
3957    \let\glsupcase\@firstofthree
3958    \let\glsinsert\@empty

```

Determine whether starred or unstarred version was used:

```

3959    \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper
3960    \setkeys{glslink}{hyper=true,#1}%
3961    \ifKV@glslink@hyper
3962      \let\glsifhyper\@firstoftwo
3963    \else
3964      \let\glsifhyper\@secondoftwo
3965    \fi
3966    \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper

```

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

```

3967    \def\glscustomtext{%
3968      \glsentrylongpl{#2}#3%
3969    }%

```

Call \@gls@link

```

3970    \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3971  }%
3972 }

```

\Acrlongpl

```

3973 \newrobustcmd*{\Acrlongpl}{\@ifstar\s@Acrlongpl\ns@Acrlongpl}

```

Define the starred form:

```

3974 \newcommand*\s@Acrlongpl[2][{}]{%
3975   \new@ifnextchar[{\@Acrlongpl{hyper=false#1}{#2}}{%
3976     {\@Acrlongpl{hyper=false,#1}{#2}[]}%
3977 }

```

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

```

3978 \newcommand*\ns@Acrlongpl[2][{}]{%
3979   \new@ifnextchar[{\@Acrlongpl{#1}{#2}}{\@Acrlongpl{#1}{#2}[]}%
3980 }

```

Read in the final optional argument:

```

3981 \def\@Acrlongpl#1#2[#3]{%
3982   \glsdoifexists{#2}%
3983   {%
3984     \edef\@glo@type{\glsentrytype{#2}}%

```



```

3985 \def\glslabel{#2}%
3986 \let\glsifplural\@firstoftwo
3987 \let\glscapscase\@secondofthree
3988 \let\glsinsert\@empty

```

Determine whether starred or unstarred version was used:

```

3989 \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper
3990 \setkeys{glslink}{hyper=true,#1}%
3991 \ifKV@glslink@hyper
3992 \let\glsifhyper\@firstoftwo
3993 \else
3994 \let\glsifhyper\@secondoftwo
3995 \fi
3996 \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper

```

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

```

3997 \def\glscustomtext{%
3998 \Glsentrylongpl{#2}#3%
3999 }%

```

Call \@gls@link

```

4000 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
4001 }%
4002 }

```

\ACRlongpl

```

4003 \newrobustcmd*{\ACRlongpl}{\@ifstar\s@ACRlongpl\@ns@ACRlongpl}

```

Define the starred form:

```

4004 \newcommand*{\s@ACRlongpl}[2][{}]{%
4005 \new@ifnextchar[{\@ACRlongpl{hyper=false,#1}{#2}}%
4006 {\@ACRlongpl{hyper=false,#1}{#2}[]}%
4007 }

```

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

```

4008 \newcommand*{\ns@ACRlongpl}[2][{}]{%
4009 \new@ifnextchar[{\@ACRlongpl{#1}{#2}}{\@ACRlongpl{#1}{#2}[]}%
4010 }

```

Read in the final optional argument:

```

4011 \def\@ACRlongpl#1#2[#3]{%
4012 \glsdoifexists{#2}%
4013 {%
4014 \edef\@glo@type{\glentrytype{#2}}%

4015 \def\glslabel{#2}%
4016 \let\glsifplural\@firstoftwo
4017 \let\glscapscase\@thirdofthree
4018 \let\glsinsert\@empty

```

Determine whether starred or unstarred version was used:

```

4019 \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper
4020 \setkeys{glslink}{hyper=true,#1}%
4021 \ifKV@glslink@hyper
4022   \let\glsifhyper\@firstoftwo
4023 \else
4024   \let\glsifhyper\@secondoftwo
4025 \fi
4026 \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper

```

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

```

4027 \def\glscustomtext{%
4028   \mfirstucMakeUppercase{\glsentrylongpl{#2}#3}%
4029 }%

```

Call `\@gls@link`

```

4030 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
4031 }%
4032 }

```

1.10.2 Displaying entry details without adding information to the glossary

These commands merely display entry information without adding entries in the associated file or having hyperlinks.

`\@gls@entry@field` Generic version.

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

```

4033 \newcommand*{\@gls@entry@field}[2]{%
4034   \csname glo@\glsdetoklabel{#1}@\#2\endcsname
4035 }

```

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

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

```

4036 \newcommand*{\@Gls@entry@field}[2]{%
4037   \letcs\@glo@text{glo@\glsdetoklabel{#1}@\#2}%
4038   \xmakefirstuc{\@glo@text}%
4039 }

```

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
4040 \newcommand*{\glsentryname}[1]{\@gls@entry@field{#1}{name}}
```

```
\Glsentryname
4041 \newrobustcmd*{\Glsentryname}[1]{%
4042   \@Gls@entry@field{#1}{name}%
4043 }
```

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
4044 \newcommand*{\glsentrydesc}[1]{\@gls@entry@field{#1}{desc}}
```

```
\Glsentrydesc
4045 \newrobustcmd*{\Glsentrydesc}[1]{%
4046   \@Gls@entry@field{#1}{desc}%
4047 }
```

Plural form:

```
\glsentrydescplural
4048 \newcommand*{\glsentrydescplural}[1]{%
4049   \@gls@entry@field{#1}{descplural}%
4050 }
```

```
\Glsentrydescplural
4051 \newrobustcmd*{\Glsentrydescplural}[1]{%
4052   \@Gls@entry@field{#1}{descplural}%
4053 }
```

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
4054 \newcommand*{\glsentrytext}[1]{\@gls@entry@field{#1}{text}}
```

```
\Glsentrytext
4055 \newrobustcmd*{\Glsentrytext}[1]{%
4056   \@Gls@entry@field{#1}{text}%
4057 }
```

Get the plural form:

```
\glsentryplural
4058 \newcommand*{\glsentryplural}[1]{%
4059   \@gls@entry@field{#1}{plural}%
4060 }
```

`\Glsentryplural`

```
4061 \newrobustcmd*{\Glsentryplural}[1]{%
4062   \@Gls@entry@field{#1}{plural}%
4063 }
```

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

`\glsentrysymbol`

```
4064 \newcommand*{\glsentrysymbol}[1]{%
4065   \@Gls@entry@field{#1}{symbol}%
4066 }
```

`\Glsentrysymbol`

```
4067 \newrobustcmd*{\Glsentrysymbol}[1]{%
4068   \@Gls@entry@field{#1}{symbol}%
4069 }
```

Plural form:

`\glsentrysymbolplural`

```
4070 \newcommand*{\glsentrysymbolplural}[1]{%
4071   \@Gls@entry@field{#1}{symbolplural}%
4072 }
```

`\Glsentrysymbolplural`

```
4073 \newrobustcmd*{\Glsentrysymbolplural}[1]{%
4074   \@Gls@entry@field{#1}{symbolplural}%
4075 }
```

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`

```
4076 \newcommand*{\glsentryfirst}[1]{%
4077   \@Gls@entry@field{#1}{first}%
4078 }
```

`\Glsentryfirst`

```
4079 \newrobustcmd*{\Glsentryfirst}[1]{%
4080   \@Gls@entry@field{#1}{first}%
4081 }
```

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

`\glsentryfirstplural`

```
4082 \newcommand*{\glsentryfirstplural}[1]{%
4083   \@Gls@entry@field{#1}{firstpl}%
4084 }
```

Glsentryfirstplural

```
4085 \newrobustcmd*{\Glsentryfirstplural}[1]{%
4086   \@Gls@entry@field{#1}{firstpl}%
4087 }
```

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

\glsentrytype

```
4088 \newcommand*{\glsentrytype}[1]{\@Gls@entry@field{#1}{type}}
```

Display the sort text used for this entry. Note that the sort key is sanitized, so unexpected results may occur if the sort key contained commands.

\glsentrysort

```
4089 \newcommand*{\glsentrysort}[1]{%
4090   \@Gls@entry@field{#1}{sort}%
4091 }
```

\glsentryuseri Get the first user key (as specified by the user1 when the entry was defined).
The argument is the label associated with the entry.

```
4092 \newcommand*{\glsentryuseri}[1]{%
4093   \@Gls@entry@field{#1}{useri}%
4094 }
```

\Glsentryuseri

```
4095 \newrobustcmd*{\Glsentryuseri}[1]{%
4096   \@Gls@entry@field{#1}{useri}%
4097 }
```

\glsentryuserii Get the second user key (as specified by the user2 when the entry was defined).
The argument is the label associated with the entry.

```
4098 \newcommand*{\glsentryuserii}[1]{%
4099   \@Gls@entry@field{#1}{userii}%
4100 }
```

\Glsentryuserii

```
4101 \newrobustcmd*{\Glsentryuserii}[1]{%
4102   \@Gls@entry@field{#1}{userii}%
4103 }
```

\glsentryuseriii Get the third user key (as specified by the user3 when the entry was defined).
The argument is the label associated with the entry.

```
4104 \newcommand*{\glsentryuseriii}[1]{%
4105   \@Gls@entry@field{#1}{useriii}%
4106 }
```

\Glsentryuseriii

```
4107 \newrobustcmd*{\Glsentryuseriii}[1]{%
4108   \@Gls@entry@field{#1}{useriii}%
4109 }
```

\glsentryuseriv Get the fourth user key (as specified by the user4 when the entry was defined).
The argument is the label associated with the entry.

```
4110 \newcommand*{\glsentryuseriv}[1]{%
4111   \@Gls@entry@field{#1}{useriv}%
4112 }
```

\Glsentryuseriv

```
4113 \newrobustcmd*{\Glsentryuseriv}[1]{%
4114   \@Gls@entry@field{#1}{useriv}%
4115 }
```

\glsentryuserv Get the fifth user key (as specified by the user5 when the entry was defined).
The argument is the label associated with the entry.

```
4116 \newcommand*{\glsentryuserv}[1]{%
4117   \@Gls@entry@field{#1}{userv}%
4118 }
```

\Glsentryuserv

```
4119 \newrobustcmd*{\Glsentryuserv}[1]{%
4120   \@Gls@entry@field{#1}{userv}%
4121 }
```

\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.

```
4122 \newcommand*{\glsentryuservi}[1]{%
4123   \@Gls@entry@field{#1}{uservi}%
4124 }
```

\Glsentryuservi

```
4125 \newrobustcmd*{\Glsentryuservi}[1]{%
4126   \@Gls@entry@field{#1}{uservi}%
4127 }
```

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

```
4128 \newcommand*{\glsentryshort}[1]{\@Gls@entry@field{#1}{short}}
```

\Glsentryshort

```
4129 \newrobustcmd*{\Glsentryshort}[1]{%
4130   \@Gls@entry@field{#1}{short}%
4131 }
```

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

```

4132 \newcommand*{\glsentryshortpl}[1]{\@gls@entry@field{#1}{shortpl}}

\Glsentryshortpl
4133 \newrobustcmd*{\Glsentryshortpl}[1]{%
4134   \@Gls@entry@field{#1}{shortpl}}%
4135 }

\glsentrylong Get the long key (as specified by the long the entry was defined). The argument
is the label associated with the entry.
4136 \newcommand*{\glsentrylong}[1]{\@gls@entry@field{#1}{long}}

\Glsentrylong
4137 \newrobustcmd*{\Glsentrylong}[1]{%
4138   \@Gls@entry@field{#1}{long}}%
4139 }

\glsentrylongpl Get the long plural key (as specified by the longplural the entry was defined).
The argument is the label associated with the entry.
4140 \newcommand*{\glsentrylongpl}[1]{\@gls@entry@field{#1}{longpl}}

\Glsentrylongpl
4141 \newrobustcmd*{\Glsentrylongpl}[1]{%
4142   \@Gls@entry@field{#1}{longpl}}%
4143 }

Short cut macros to access full form:

\glsentryfull
4144 \newcommand*{\glsentryfull}[1]{%
4145   \acrfullformat{\glsentrylong{#1}}{\acronymfont{\glsentryshort{#1}}}%
4146 }

\Glsentryfull
4147 \newrobustcmd*{\Glsentryfull}[1]{%
4148   \acrfullformat{\Glsentrylong{#1}}{\acronymfont{\glsentryshort{#1}}}%
4149 }

\glsentryfullpl
4150 \newcommand*{\glsentryfullpl}[1]{%
4151   \acrfullformat{\glsentrylongpl{#1}}{\acronymfont{\glsentryshortpl{#1}}}%
4152 }

\Glsentryfullpl
4153 \newrobustcmd*{\Glsentryfullpl}[1]{%
4154   \acrfullformat{\Glsentrylongpl{#1}}{\acronymfont{\glsentryshortpl{#1}}}%
4155 }

```

`\glsentrynumberlist` Displays the number list as is.

```

4156 \newcommand*\glsentrynumberlist}[1]{%
4157   \glsdoifexists{#1}%
4158   {%
4159     \@gls@entry@field{#1}{numberlist}%
4160   }%
4161 }
```

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

```

4162 \@ifpackageloaded{hyperref} {%
4163   \newcommand*\glsdisplaynumberlist}[1]{%
4164     \GlossariesWarning
4165     {%
4166       \string\glsdisplaynumberlist\space
4167       doesn't work with hyperref.^^JUsing
4168       \string\glsentrynumberlist\space instead%
4169     }%
4170     \glsentrynumberlist{#1}%
4171   }%
4172 }%
4173 {%
4174   \newcommand*\glsdisplaynumberlist}[1]{%
4175     \glsdoifexists{#1}%
4176     {%
4177       \bgroup

4178       \edef\@glo@label{\glsdetoklabel{#1}}%
4179       \let\@org@glsglnumberformat\glsglnumberformat
4180       \def\glsglnumberformat##1{##1}%
4181       \protected@edef\the@numberlist{%
4182         \csname glo@\@glo@label @numberlist\endcsname}%
4183       \def\@glsgl@numlist@sep{}%
4184       \def\@glsgl@numlist@nextsep{}%
4185       \def\@glsgl@numlist@lastsep{}%
4186       \def\@glsgl@thislist{}%
4187       \def\@glsgl@donext@def{}%
4188       \renewcommand\do[1]{%
4189         \protected@edef\@glsgl@thislist{%
4190           \@glsgl@thislist
4191           \noexpand\@glsgl@numlist@sep
4192           ##1%
4193         }%
4194         \let\@glsgl@numlist@sep\@glsgl@numlist@nextsep
4195         \def\@glsgl@numlist@nextsep{\glsgl@numlist@sep}%
4196         \@glsgl@donext@def
4197         \def\@glsgl@donext@def{%
4198           \def\@glsgl@numlist@lastsep{\glsgl@numlist@lastsep}%
4199         }%
4200       }%

```



```

4201         \expandafter \glsnumlistparser \expandafter{\the@numberlist}%
4202         \let\@gls@numlist@sep\@gls@numlist@lastsep
4203         \@gls@thislist
4204     \egroup
4205 }%
4206 }
4207 }

```

`\glsnumlistsep`

```
4208 \newcommand*{\glsnumlistsep}{, }
```

`\glsnumlistlastsep`

```
4209 \newcommand*{\glsnumlistlastsep}{ \& }
```

`\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.

```

4210 \newcommand*{\glshyperlink}[2][\glsentrytext{\@glo@label}]{%
4211   \def\@glo@label{#2}%
4212   \@glslink{\gloslinkprefix\glsdetoklabel{#2}}{#1}}

```

1.11 Adding an entry to the glossary without generating text

The following keys are provided for `\glsadd` and `\glsaddall`:

```
4213 \define@key{glossadd}{counter}{\def\@gls@counter{#1}}
```

```
4214 \define@key{glossadd}{format}{\def\@glsnumberformat{#1}}
```

This key is only used by `\glsaddall`:

```
4215 \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`

```

4216 \newrobustcmd*{\glsadd}[2][ ]{%
4217   \glsdoifexists{#2}%
4218   {%
4219     \def\@glsnumberformat{glsnumberformat}%
4220     \edef\@gls@counter{\csname glo@glsdetoklabel{#2}@counter\endcsname}%
4221     \setkeys{glossadd}{#1}%

```

Store the entry's counter in `\theglsentrycounter`

```
4222 \gls@saveentrycounter
4223 \do@wrglossary{#2}%
4224 }%
4225 }
```

`\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`

```
4226 \newrobustcmd*{\glsaddall}[1] [] {%
4227 \edef\@glo@type{\@glo@types}%
4228 \setkeys{glossadd}{#1}%
4229 \forallglsentries[\@glo@type]{\@glo@entry}{%
4230 \glsadd[#1]{\@glo@entry}%
4231 }%
4232 }
```

`\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.

```
4233 \newrobustcmd*{\glsaddallunused}[1] [\@glo@types] {%
4234 \forallglsentries[#1]{\@glo@entry}%
4235 {%
4236 \ifglsused{\@glo@entry}{\@glsadd[format=@gobble]{\@glo@entry}}%
4237 }%
4238 }
```

1.12 Creating associated files

The `\writeist` command creates the associated customized `.ist` `makeindex` style file. While defining this command, some characters have their catcodes temporarily changed to ensure they get written to the `.ist` file correctly. The `makeindex` actual character (usually `@`) is redefined to be a `?`, to allow internal commands to be written to the glossary file output file.

The special characters are stored in `\@gls@actualchar`, `\@gls@encapchar`, `\@gls@levelchar` and `\@gls@quotechar` to make them easier to use later, but don't change these values, because the characters are encoded in the command definitions that are used to escape the special characters (which means that the user no longer needs to worry about `makeindex` special characters).

The symbols and numbers label for group headings are hardwired into the .ist file as glssymbols and glsnumbers, the group titles can be translated (so that \glssymbolsgroupname replaces glssymbols and \glsnumbersgroupname replaces glsnumbers) using the command \glssetgrouptitle which is defined in . This is done to prevent any problem characters in \glssymbolsgroupname and \glsnumbersgroupname from breaking hyperlinks.

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

```
4239 \edef\glsopenbrace{\expandafter\@gobble\string\{}
```

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

```
4240 \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.

```
4241 \edef\glsquote#1{\string"#1\string"}
```

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

```
4242 \ifglsxindy
4243   \newcommand*{\@glsfirstletter}{A}
4244 \fi
```

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

```
4245 \ifglsxindy
4246   \newcommand*{\GlsSetXdyFirstLetterAfterDigits}[1]{%
4247     \renewcommand*{\@glsfirstletter}{#1}}
4248 \else
4249   \newcommand*{\GlsSetXdyFirstLetterAfterDigits}[1]{%
4250     \glsnoxywarning\GlsSetXdyFirstLetterAfterDigits}
4251 \fi
```

`\@glsminrange` Define the minimum number of successive location references to merge into a range.

```
4252 \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.

```
4253 \ifglsxindy
4254   \newcommand*{\GlsSetXdyMinRangeLength}[1]{%
4255     \renewcommand*{\@glsminrange}{#1}}
4256 \else
4257   \newcommand*{\GlsSetXdyMinRangeLength}[1]{%
4258     \glsnoxywarning\GlsSetXdyMinRangeLength}
4259 \fi
```

`\writeist`

```
4260 \ifglsxindy
```

Code to use if xindy is required.

```

4261 \def\writeist{%
    Define write register if not already defined
4262 \ifundef{\glswrite}{\newwrite\glswrite}{}%
    Update attributes list
4263 \@gls@addpredefinedattributes
    Open the file.
4264 \openout\glswrite=\istfilename
    Write header comment at the start of the file
4265 \write\glswrite{;; xindy style file created by the glossaries
4266 package}%
4267 \write\glswrite{;; for document '\jobname' on
4268 \the\year-\the\month-\the\day}%
    Specify the required styles
4269 \write\glswrite{^^J; required styles^^J}
4270 \@for\@xdystyle:=\@xdyrequiredstyles\do{%
4271 \ifx\@xdystyle\@empty
4272 \else
4273 \protected@write\glswrite{{(require
4274 \string"\@xdystyle.xdy\string")}}%
4275 \fi
4276 }%
    List the allowed attributes (possible values used by the format key)
4277 \write\glswrite{^^J%
4278 ; list of allowed attributes (number formats)^^J}%
4279 \write\glswrite{(define-attributes ((\@xdyattributes)))}%
    Define any additional alphabets
4280 \write\glswrite{^^J; user defined alphabets^^J}%
4281 \write\glswrite{\@xdyuseralphabets}%
    Define location classes.
4282 \write\glswrite{^^J; location class definitions^^J}%
    As from version 3.0, locations are now specified as {\<Hprefix>}{\<number>}, so
    need to add all possible combinations of location types.
4283 \@for\@gls@classI:=\@gls@xdy@locationlist\do{%
    Case were \<Hprefix> is empty:
4284 \protected@write\glswrite{{(define-location-class
4285 \string"\@gls@classI\string"^^J\space\space\space
4286 (
4287 :sep "{}{"
4288 \csname @gls@xdy@Lclass@\@gls@classI\endcsname\space
4289 :sep "}"
4290 )
4291 ^^J\space\space\space

```

```

4292         :min-range-length \@glsminrange^^J%
4293     )
4294 }%

```

Nested iteration over all classes:

```

4295     {%
4296         \@for\@gls@classII:=\@gls@xdy@locationlist\do{%
4297             \protected@write\glswrite{}\{(define-location-class
4298                 \string"\@gls@classII-\@gls@classI\string"
4299                 ^^J\space\space\space
4300                 (
4301                     :sep "{"
4302                     \csname \@gls@xdy@Lclass@\@gls@classII\endcsname\space
4303                     :sep "{{"
4304                     \csname \@gls@xdy@Lclass@\@gls@classI\endcsname\space
4305                     :sep "}"
4306                 )
4307                 ^^J\space\space\space
4308                 :min-range-length \@glsminrange^^J%
4309             )
4310         }%
4311     }%
4312 }%
4313 }%

```

User defined location classes (needs checking for new location format).

```

4314 \write\glswrite{^^J; user defined location classes}%
4315 \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.)

```

4316 \write\glswrite{^^J; define cross-reference class^^J}%
4317 \write\glswrite{(define-crossref-class \string"see\string"
4318     :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.)

```

4319 \write\glswrite{(markup-crossref-list
4320     :class \string"see\string"^^J\space\space\space
4321     :open \string"\string\glsseeformat\string"
4322     :close \string"{}\string")}%

```

List the order to sort the classes.

```

4323 \write\glswrite{^^J; define the order of the location classes}%
4324 \write\glswrite{(define-location-class-order
4325     (\@xdylocationclassorder))}%

```

Specify what to write to the start and end of the glossary file.

```

4326 \write\glswrite{^^J; define the glossary markup^^J}%

```

```

4327 \write\glswrite{(markup-index^^J\space\space\space
4328 :open \string"\string
4329 \glossarysection[\string\glossarytoctitle]{\string
4330 \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)

```

4331 \@for\@this@ctr:=\@xdycounters\do{%
4332 {%
4333 \for\@this@attr:=\@xdyattributelist\do{%
4334 \protected\write\glswrite{ }\string\providecommand*%
4335 \expandafter\string
4336 \csname glsX\@this@ctr X\@this@attr\endcsname[2]%
4337 {%
4338 \string\setentrycounter
4339 [\expandafter\@gobble\string\#1]{\@this@ctr}%
4340 \expandafter\string
4341 \csname\@this@attr\endcsname
4342 {\expandafter\@gobble\string\#2}%
4343 }%
4344 }%
4345 }%
4346 }%
4347 }%

```

Add the end part of the open tag and the rest of the markup-index information:

```

4348 \write\glswrite{%
4349 \string\begin
4350 {theglossary}\string\glossaryheader\string~n\string" ^^J\space
4351 \space\space:close \string"\expandafter\@gobble
4352 \string~n\string
4353 \end{theglossary}\string\glossarypostamble
4354 \string~n\string" ^^J\space\space\space
4355 :tree)}}%

```

Specify what to put between letter groups

```

4356 \write\glswrite{(markup-letter-group-list
4357 :sep \string"\string\glsgrpskip\string~n\string"))}%

```

Specify what to put between entries

```

4358 \write\glswrite{(markup-indexentry
4359 :open \string"\string\relax \string\glsresetentrylist
4360 \string~n\string"))}%

```

Specify how to format entries

```

4361 \write\glswrite{(markup-locclass-list :open
4362 \string"\glsopenbrace\string\glossaryentrynumbers
4363 \glsopenbrace\string\relax\space \string"^^J\space\space\space
4364 :sep \string", \string"
4365 :close \string"\glsclosebrace\glsresetentrylist\string"))}%

```

Specify how to separate location numbers

```
4366 \write\glswrite{(markup-locref-list
4367 :sep \string\string\delimN\space\string)}}%
```

Specify how to indicate location ranges

```
4368 \write\glswrite{(markup-range
4369 :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.

```
4370 \@onelevel@sanitize\gls@suffiF
4371 \@onelevel@sanitize\gls@suffiFF

4372 \ifx\gls@suffiF\@empty
4373 \else
4374 \write\glswrite{(markup-range
4375 :close "\gls@suffiF" :length 1 :ignore-end)}}%
4376 \fi
4377 \ifx\gls@suffiFF\@empty
4378 \else
4379 \write\glswrite{(markup-range
4380 :close "\gls@suffiFF" :length 2 :ignore-end)}}%
4381 \fi
```

Specify how to format locations.

```
4382 \write\glswrite{^^J; define format to use for locations^^J}%
4383 \write\glswrite{\@xdylocref}}%
```

Specify how to separate letter groups.

```
4384 \write\glswrite{^^J; define letter group list format^^J}%
4385 \write\glswrite{(markup-letter-group-list
4386 :sep \string\string\glsgroupskip\string~n\string)}}%
```

Define letter group headings.

```
4387 \write\glswrite{^^J; letter group headings^^J}%
4388 \write\glswrite{(markup-letter-group
4389 :open-head \string\string\glsgroupheading
4390 \glsopenbrace\string^^J\space\space\space
4391 :close-head \string\glsclosebrace\string)}}%
```

Define additional letter groups.

```
4392 \write\glswrite{^^J; additional letter groups^^J}%
4393 \write\glswrite{\@xdylettergroups}}%
```

Define additional sort rules

```
4394 \write\glswrite{^^J; additional sort rules^^J}
4395 \write\glswrite{\@xdysortrules}}%
```

Close the style file

```
4396 \closeout\glswrite
```

Suppress any further calls.

```
4397 \let\writeist\relax
4398 }
4399 \else
```

Code to use if makeindex is required.

```
4400 \edef\@gls@actualchar{\string?}
4401 \edef\@gls@encapchar{\string|}
4402 \edef\@gls@levelchar{\string!}
4403 \edef\@gls@quotechar{\string"}
4404 \def\writeist{\relax
4405 \ifundef{glswrite}{\newwrite\glswrite}{\relax
4406 \openout\glswrite=\istfilename
4407 \write\glswrite{\expandafter\@gobble\string\% makeindex style file
4408 created by the glossaries package}
4409 \write\glswrite{\expandafter\@gobble\string\% for document
4410 '\jobname' on \the\year-\the\month-\the\day}
4411 \write\glswrite{actual '\@gls@actualchar'}
4412 \write\glswrite{encap '\@gls@encapchar'}
4413 \write\glswrite{level '\@gls@levelchar'}
4414 \write\glswrite{quote '\@gls@quotechar'}
4415 \write\glswrite{keyword \string"\string\glossaryentry\string"}
4416 \write\glswrite{preamble \string"\string\glossarysection[\string
4417 \glossarytoctitle]{\string\glossarytitle}\string
4418 \glossarypreamble\string\n\string\begin{theglossary}\string
4419 \glossaryheader\string\n\string"}
4420 \write\glswrite{postamble \string"\string%\string\n\string
4421 \end{theglossary}\string\glossarypostamble\string\n
4422 \string"}
4423 \write\glswrite{group_skip \string"\string\glsgroupskip\string\n
4424 \string"}
4425 \write\glswrite{item_0 \string"\string%\string\n\string"}
4426 \write\glswrite{item_1 \string"\string%\string\n\string"}
4427 \write\glswrite{item_2 \string"\string%\string\n\string"}
4428 \write\glswrite{item_01 \string"\string%\string\n\string"}
4429 \write\glswrite{item_x1
4430 \string"\string\relax \string\glsresetentrylist\string\n
4431 \string"}
4432 \write\glswrite{item_12 \string"\string%\string\n\string"}
4433 \write\glswrite{item_x2
4434 \string"\string\relax \string\glsresetentrylist\string\n
4435 \string"}

4436 \write\glswrite{delim_0 \string"\string\{\string
4437 \glossaryentrynumbers\string\{\string\relax \string"}
4438 \write\glswrite{delim_1 \string"\string\{\string
4439 \glossaryentrynumbers\string\{\string\relax \string"}
4440 \write\glswrite{delim_2 \string"\string\{\string
4441 \glossaryentrynumbers\string\{\string\relax \string"}
4442 \write\glswrite{delim_t \string"\string\}\string\}\string"}
```



```

4443 \write\glswrite{delim_n \string"\string\\delimN \string"}
4444 \write\glswrite{delim_r \string"\string\\delimR \string"}
4445 \write\glswrite{headings_flag 1}
4446 \write\glswrite{heading_prefix
4447   \string"\string\glsgroupheading\string\{\string"}
4448 \write\glswrite{heading_suffix
4449   \string"\string\}\string\relax
4450   \string\glsgroupresetentrylist \string"}
4451 \write\glswrite{symhead_positive \string"glssymbols\string"}
4452 \write\glswrite{numhead_positive \string"glsglnumbers\string"}
4453 \write\glswrite{page_compositor \string"glsglcompositor\string"}
4454 \@gls@escbsdq\gls@suffixF
4455 \@gls@escbsdq\gls@suffixFF
4456 \ifx\gls@suffixF\@empty
4457 \else
4458   \write\glswrite{suffix_2p \string"\gls@suffixF\string"}
4459 \fi
4460 \ifx\gls@suffixFF\@empty
4461 \else
4462   \write\glswrite{suffix_3p \string"\gls@suffixFF\string"}
4463 \fi
4464 \closeout\glswrite
4465 \let\writeist\relax
4466 }
4467 \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`

```

4468 \newcommand{\noist}{%
  Update attributes list
4469   \@gls@addpredefinedattributes
4470   \let\writeist\relax
4471 }

```

`\@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`

```

4472 \newcommand*{\@makeglossary}[1]{%
4473   \ifglossaryexists{#1}%
4474   {%

```

Only create a new write if savewrites=false otherwise create a token to collect the information.

```

4475   \ifglssavewrites
4476     \expandafter\newtoks\csname glo@#1@filetok\endcsname
4477   \else
4478     \expandafter\newwrite\csname glo@#1@file\endcsname
4479     \expandafter\@glsopenfile\csname glo@#1@file\endcsname{#1}%
4480   \fi
4481   \@gls@renewglossary
4482   \writeist
4483 }%
4484 {%
4485   \PackageError{glossaries}%
4486   {Glossary type ‘#1’ not defined}%
4487   {New glossaries must be defined before using \string\makeglossary}%
4488 }%
4489 }

```

`\@glsopenfile` Open write file associated with the given glossary.

```

4490 \newcommand*{\@glsopenfile}[2]{%
4491   \immediate\openout#1=\jobname.\csname @glotype@#2@out\endcsname
4492   \PackageInfo{glossaries}{Writing glossary file
4493     \jobname.\csname @glotype@#2@out\endcsname}%
4494 }

```

`\@nomakeglossaries` Issue warning that `\makeglossaries` hasn't been used.

```

4495 \newcommand*{\@warn@nomakeglossaries}{}

```

Only use this if warning if `\printglossary` has been used without `\makeglossaries`

```

4496 \newcommand*{\@warn@nomakeglossaries}{\@warn@nomakeglossaries}

```

`\makeglossaries` will use `\@makeglossary` for each glossary type that has been defined. New glossaries need to be defined before using `\makeglossary`, so have `\makeglossaries` redefine `\newglossary` to prevent it being used afterwards.

`\makeglossaries`

```

4497 \newcommand*{\makeglossaries}{%

```

Define the write used for style file also used for all other output files if savewrites=true.

```

4498   \ifundef{glswrite}{\newwrite\glswrite}{}%

```

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

```

4499 \protected@write\@auxout{}\string\providecommand\string\@glsorder[1]{}
4500 \protected@write\@auxout{}\string\providecommand\string\@istfilename[1]{}

```

Write the name of the style file to the aux file (needed by makeglossaries)

```

4501 \protected@write\@auxout{}\string\@istfilename{\istfilename}}%
4502 \protected@write\@auxout{}\string\@glsorder{\glsorder}}

```

Iterate through each glossary type and activate it.

```

4503 \@for\@glo@type:=\@glo@types\do{%
4504   \ifthenelse{\equal{\@glo@type}{}}{}{%
4505     \@makeglossary{\@glo@type}}%
4506   }%

```

New glossaries must be created before \makeglossaries so disable \newglossary.

```

4507 \renewcommand*\newglossary[4][]{%
4508   \PackageError{glossaries}{New glossaries
4509     must be created before \string\makeglossaries}{You need
4510     to move \string\makeglossaries\space after all your
4511     \string\newglossary\space commands}}%

```

Any subsequence instances of this command should have no effect

```

4512 \let\@makeglossary\relax
4513 \let\makeglossary\relax
4514 \let\makeglossaries\relax

```

Disable all commands that have no effect after \makeglossaries

```

4515 \@disable@onlypremakeg

```

Allow see key:

```

4516 \let\gls@checkseeallowed\relax

```

Suppress warning about no \makeglossaries

```

4517 \let\warn@nomakeglossaries\relax

```

Activate warning about missing \printglossary

```

4518 \def\warn@noprintglossary{%
4519   \GlossariesWarningNoLine{No \string\printglossary\space
4520     or \string\printglossaries\space
4521     found.^^J(Remove \string\makeglossaries\space if you don't want
4522     any glossaries.)^^JThis document will not have a glossary}%
4523 }%

```

Declare list parser for \glsdisplaynumberlist

```

4524 \ifglssavenumberlist
4525   \edef\@gls@doddeflistparser{\noexpand\DeclareListParser
4526     {\noexpand\glsnumlistparser}{\delimN}}%
4527   \@gls@doddeflistparser
4528 \fi

```

Prevent user from also using \makenoidxglossaries

```

4529 \let\makenoidxglossaries\@no@makeglossaries

```

Prohibit sort key in printgloss family:

```
4530 \renewcommand*{\@printgloss@setsort}{%
4531 \let\@glo@assign@sortkey\@glo@no@assign@sortkey
4532 }%
4533 }
```

Must occur in the preamble:

```
4534 \@onlypreamble{\makeglossaries}
```

`\glswrite` The definition of `\glswrite` has now been moved to `\makeglossaries` so that it's only defined if needed.

The `\makeglossary` command is redefined to be identical to `\makeglossaries`. (This is done to reinforce the message that you must either use `\@makeglossary` for all the glossaries or for none of them.)

`\makeglossary`

```
4535 \let\makeglossary\makeglossaries
```

If `\makeglossaries` hasn't been used, issue a warning. Also issue a warning if neither `\printglossaries` nor `\printglossary` have been used.

```
4536 \AtEndDocument{%
4537 \warn@nomakeglossaries
4538 \warn@noprintglossary
4539 }
```

`\makenoidxglossaries` Analogous to `\makeglossaries` this activates the commands needed for `\printnoidxglossary`

```
4540 \newcommand*{\makenoidxglossaries}{%
```

Redefine empty glossary warning:

```
4541 \renewcommand{\@gls@noref@warn}[1]{%
4542 \GlossariesWarning{Empty glossary for
4543 \string\printnoidxglossary[type={##1}].
4544 Rerun may be required (or you may have forgotten to use
4545 commands like \string\gls).}%
4546 }
```

Don't escape makeindex/xindy characters

```
4547 \let\@gls@checkmkidxchars\@gobble
```

Write glossary information to aux instead of glossary files

```
4548 \let\@@do@@wrglossary\gls@noidxglossary
```

Switch on group headings that use the character code:

```
4549 \let\@gls@getgrouptitle\@gls@noidx@getgrouptitle
```

Allow see key:

```
4550 \let\gls@checkseeallowed\relax
```

Redefine cross-referencing macro:

```
4551 \renewcommand{\@do@seeglossary}[2]{%
4552   \edef\@gls@label{\glsdetoklabel{##1}}%
4553   \protected@write\@auxout{}\@auxout{\@gls@label}%
4554   \string\@gls@reference
4555   {\csname glo@\@gls@label @type\endcsname}%
4556   {\@gls@label}%
4557   {%
4558     \string\glsseeformat##2}%
4559   }%
4560 }%
4561 }
```

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

```
4562 \AtBeginDocument
4563 {%
4564   \write\@auxout{\string\providecommand\string\@gls@reference[3]{}}%
4565 }
```

Change warning about no glossaries

```
4566 \def\warn@noprintglossary{%
4567   \GlossariesWarningNoLine{No \string\printnoidxglossary\space
4568     or \string\printnoidxglossaries ^^J
4569     found. (Remove \string\makenoidxglossaries\space if you
4570     don't want any glossaries.)^^JThis document will not have a glossary}%
4571 }
```

Suppress warning about no \makeglossaries

```
4572 \let\warn@nomakeglossaries\relax
```

Prevent user from also using \makeglossaries

```
4573 \let\makeglossaries\@no@makeglossaries
```

Allow sort key in printgloss family:

```
4574 \renewcommand*\@printgloss@setsort}{%
4575   \let\@glo@assign@sortkey\@glo@assign@sortkey
```

Initialise default sort order:

```
4576 \def\@glo@sorttype{\@glo@default@sorttype}%
4577 }
```

All entries must be defined in the preamble:

```
4578 \renewcommand*\new@glossaryentry[2]{%
4579   \PackageError{glossaries}{Glossary entries must be
4580     defined in the preamble^^Jwhen you use
4581     \string\makenoidxglossaries}%
4582   {Either move your definitions to the preamble or use
4583     \string\makeglossaries}%
4584 }
```

```

Redefine \glsentrynumberlist
4585 \renewcommand*{\glsentrynumberlist}[1]{%
4586 \letcs{\@gls@loclist}{glo@\glsdetoklabel{##1}@loclist}%
4587 \ifdef\@gls@loclist
4588 {%
4589 \glsnoidxloclist{\@gls@loclist}%
4590 }%
4591 {%
4592 \ifglsentryexists{##1}%
4593 {%
4594 \GlossariesWarning{Missing location list for ‘##1’. Either
4595 a rerun is required or you haven’t referenced the entry.}%
4596 }%
4597 {%
4598 \PackageError{glossaries}{Glossary entry ‘##1’ has not been
4599 defined.}{}%
4600 }%
4601 }%
4602 }%

Redefine \glsdisplaynumberlist
4603 \renewcommand*{\glsdisplaynumberlist}[1]{%
4604 \letcs{\@gls@loclist}{glo@\glsdetoklabel{##1}@loclist}%
4605 \ifdef\@gls@loclist
4606 {%
4607 \def\@gls@noidxloclist@sep{%
4608 \def\@gls@noidxloclist@sep{%
4609 \def\@gls@noidxloclist@sep{%
4610 \glsnumlistsep
4611 }%
4612 \def\@gls@noidxloclist@finalsep{\glsnumlistlastsep}%
4613 }%
4614 }%
4615 \def\@gls@noidxloclist@finalsep{}%
4616 \def\@gls@noidxloclist@prev{}%
4617 \forlistloop{\glsnoidxdisplayloclisthandler}{\@gls@loclist}%
4618 \@gls@noidxloclist@finalsep
4619 \@gls@noidxloclist@prev
4620 }%
4621 {%
4622 ??\ifglsentryexists{##1}%
4623 {%
4624 \GlossariesWarning{Missing location list for ‘##1’. Either
4625 a rerun is required or you haven’t referenced the entry.}%
4626 }%
4627 {%
4628 \PackageError{glossaries}{Glossary entry ‘##1’ has not been
4629 defined.}{}%
4630 }%
4631 }%

```

4632 }%

Provide a generic way of iterating through the number list:

```

4633 \renewcommand*\glsnumberlistloop}[3]{%
4634   \letcs{\@gls@loclist}{glo@glstdetoklabel{##1}@loclist}%
4635   \let\@gls@org@glsnoidxdisplayloc\glsnoidxdisplayloc
4636   \let\@gls@org@glssseeformat\glssseeformat
4637   \let\glsnoidxdisplayloc##2\relax
4638   \let\glssseeformat##3\relax
4639   \ifdef\@gls@loclist
4640   {%
4641     \forlistloop{\glsnoidxnumberlistloophandler}{\@gls@loclist}%
4642   }%
4643   {%
4644     \ifglsentryexists{##1}%
4645     {%
4646       \GlossariesWarning{Missing location list for ‘##1’. Either
4647         a rerun is required or you haven’t referenced the entry.}%
4648     }%
4649     {%
4650       \PackageError{glossaries}{Glossary entry ‘##1’ has not been
4651         defined.}{}%
4652     }%
4653   }%
4654   \let\glsnoidxdisplayloc\@gls@org@glsnoidxdisplayloc
4655   \let\glssseeformat\@gls@org@glssseeformat
4656 }%
```

Modify sanitize sort function

```

4657 \let\@gls@sanitizesort\@gls@noidx@sanitizesort
4658 \let\@gls@nosanitizesort\@gls@noidx@nosanitizesort
4659 \@gls@noidx@setsanitizesort
4660 }
```

Preamble-only command:

```

4661 \@onlypreamble{\makenoidxglossaries}
```

\glsnumberlistloop \glsnumberlistloop{<label>}{<handler>}

```

4662 \newcommand*\glsnumberlistloop}[2]{%
4663   \PackageError{glossaries}{\string\glsnumberlistloop\space
4664     only works with \string\makenoidxglossaries}{}%
4665 }
```

numberlistloophandler Handler macro for \glsnumberlistloop. (The argument should be in the form \glsnoidxdisplayloc{<prefix>}{<counter>}{<format>}{<n>})

```

4666 \newcommand*\glsnoidxnumberlistloophandler}[1]{%
4667   #1%
4668 }
```

```

\@no@makeglossaries Can't use both \makeglossaries and \makenoidxglossaries
4669 \newcommand*{\@no@makeglossaries}{%
4670   \PackageError{glossaries}{You can't use both
4671     \string\makeglossaries\space and \string\makenoidxglossaries}%
4672   {Either use one or other (or none) of those commands but not both
4673     together.}%
4674 }

\@gls@noref@warn Warning when no instances of \@gls@reference found.
4675 \newcommand{\@gls@noref@warn}[1]{%
4676   \GlossariesWarning{\string\makenoidxglossaries\space
4677     is required to make \string\printnoidxglossary[type={#1}] work}%
4678 }

\gls@noidxglossary Write the glossary information to the aux file:
4679 \newcommand*{\gls@noidxglossary}{%
4680   \protected@write\@auxout{}{%
4681     \string\@gls@reference
4682     {\csname glo@\@gls@label @type\endcsname}%
4683     {\@gls@label}%
4684     {\string\glsnoidxdisplayloc
4685       {\@glo@counterprefix}%
4686       {\@gls@counter}%
4687       {\@glsnumberformat}%
4688       {\@glslocref}%
4689     }%
4690   }%
4691 }

```

1.13 Writing information to associated files

`\istfile` Deprecated.

```
4692 \def\istfile{\glswrite}
```

At the end of the document, the files should be created if `savewrites=true`.

```

4693 \AtEndDocument{%
4694   \glswritefiles
4695 }

```

`\@glswritefiles` Only write the files if `savewrites=true`

```

4696 \newcommand*{\@glswritefiles}{%
  Iterate through all the glossaries
4697   \forallglossaries{\@glo@type}{%
    Check for empty glossaries (patch provided by Patrick Häcker)
4698     \ifcsundef{glo@\@glo@type @filetok}%
4699     {%
4700       \def\gls@tmp{%

```



```

4701 }%
4702 {%
4703     \edef\gls@tmp{\expandafter\the
4704         \csname glo@\@glo@type @filetok\endcsname}%
4705 }%
4706 \ifx\gls@tmp\@empty
4707     \ifx\@glo@type\glsdefaulttype
4708         \GlossariesWarningNoLine{Glossary '@@glo@type' has no
4709             entries.^^JRemember to use package option 'nomain' if
4710 you
4711             don't want to^^Juse the main glossary}%
4712     \else
4713         \GlossariesWarningNoLine{Glossary '@@glo@type' has no
4714             entries}%
4715     \fi
4716 \else
4717     \@glsopenfile{\glswrite}{\@glo@type}%
4718     \immediate\write\glswrite{%
4719         \expandafter\the
4720         \csname glo@\@glo@type @filetok\endcsname}%
4721     \immediate\closeout\glswrite
4722 \fi
4723 }%
4724 }

```

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`

```

4725 \renewcommand*{\glossary}[1][\glsdefaulttype]{%
4726     \@glossary[#1]%
4727 }

```

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`

```

4728 \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`

```

4729 \newcommand{\@gls@renewglossary}{%
4730     \gdef\@glossary[#1]{\@bsphack\begingroup\@wrglossary{##1}}%

```

```

4731 \let\@gls@renewglossary\@empty
4732 }

```

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`

```

4733 \renewcommand*{\@wrglossary}[2]{%
4734   \ifglssavewrites
4735     \protected@edef\@gls@tmp{\the\csname glo@#1@filetok\endcsname#2}%
4736     \expandafter\global\expandafter\csname glo@#1@filetok\endcsname
4737       \expandafter{\@gls@tmp^^J}%
4738   \else
4739     \ifcsdef{glo@#1@file}%
4740     {%
4741       \expandafter\protected@write\csname glo@#1@file\endcsname{%
4742         \gls@disablepagerefexpansion}{#2}%
4743     }%
4744     {%
4745       \GlossariesWarning{No file defined for glossary ‘#1’}%
4746     }%
4747   \fi
4748 \endgroup\@esphack
4749 }

```

`\@do@wrglossary`

```

4750 \newcommand*{\@do@wrglossary}[1]{%
4751   \ifglsindexonlyfirst
4752     \ifglsused{#1}{\{\@do@wrglossary{#1}\}}%
4753   \else
4754     \@do@wrglossary{#1}%
4755   \fi
4756 }

```

`@protected@pagefmts` List of page formats to be protected against expansion.

```

4757 \newcommand{\gls@protected@pagefmts}{%
4758   \gls@numberpage,\gls@alphpage,\gls@Alphpage,\gls@romanpage,\gls@Romanpage%
4759 }

```

`blepagerefexpansion`

```

4760 \newcommand*{\gls@disablepagerefexpansion}{%
4761   \@for\@gls@this:=\gls@protected@pagefmts\do
4762   {%
4763     \expandafter\let\@gls@this\relax
4764   }%
4765 }

```

```

\gls@alphpage
4766 \newcommand*{\gls@alphpage}{\@alph\c@page}

\gls@Alphpage
4767 \newcommand*{\gls@Alphpage}{\@Alph\c@page}

\gls@numberpage
4768 \newcommand*{\gls@numberpage}{\number\c@page}

\gls@romanpage
4769 \newcommand*{\gls@romanpage}{\romannumeral\c@page}

\gls@Romanpage
4770 \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.
4771 \newcommand*{\@do@wrglossary}[1]{%
4772   \begingroup
      First a bit of hackery to prevent premature expansion of \c@page. Store original
      definitions:
4773   \let\orgthe\the
4774   \let\orgnumber\number
4775   \let\orgromannumeral\romannumeral
4776   \let\orgalph\@alph
4777   \let\orgAlph\@Alph
4778   \let\orgRoman\@Roman
      Redefine:
4779   \def\the##1{%
4780     \ifx##1\c@page \gls@numberpage\else\orgthe##1\fi}%
4781   \def\number##1{%
4782     \ifx##1\c@page \gls@numberpage\else\orgnumber##1\fi}%
4783   \def\romannumeral##1{%
4784     \ifx##1\c@page \gls@romanpage\else\orgromannumeral##1\fi}%
4785   \def\@Roman##1{%
4786     \ifx##1\c@page \gls@Romanpage\else\orgRoman##1\fi}%
4787   \def\@alph##1{%
4788     \ifx##1\c@page \gls@alphpage\else\orgalph##1\fi}%
4789   \def\@Alph##1{%
4790     \ifx##1\c@page \gls@Alphpage\else\orgAlph##1\fi}%
      Prevent expansion:
4791   \gls@disablepagerefexpansion
      Now store location in \glslocref:
4792   \protected@xdef\glslocref{\theglsentrycounter}%
4793   \endgroup

```

Escape any special characters

```
4794 \@gls@checkmkidxchars\@glslocref
```

Check if the hyper-location is the same as the location and set the hyper prefix.

```
4795 \expandafter\ifx\theHglentrycounter\theHglentrycounter\relax
4796 \def\@glo@counterprefix{}%
4797 \else
4798 \protected@edef\@glsHlocref{\theHglentrycounter}%
4799 \@gls@checkmkidxchars\@glsHlocref
4800 \edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix
4801 {\@glslocref}{\@glsHlocref}%
4802 }%
4803 \@do@gls@getcounterprefix
4804 \fi
```

De-tok label if required

```
4805 \edef\@gls@label{\glsdetoklabel{#1}}%
```

Write the information to file:

```
4806 \@do@wrglossary
4807 }
```

```
\@do@wrglossary
```

```
4808 \newcommand*{\@do@wrglossary}{%
```

Determine whether to use xindy or makeindex syntax

```
4809 \ifglxindy
```

Need to determine if the formatting information starts with a (or) indicating a range.

```
4810 \expandafter\@glo@check@mkidxrangechar\@glsnumberformat\@nil
4811 \def\@glo@range{}%
4812 \expandafter\if\@glo@prefix(\relax
4813 \def\@glo@range{:open-range}%
4814 \else
4815 \expandafter\if\@glo@prefix)\relax
4816 \def\@glo@range{:close-range}%
4817 \fi
4818 \fi
```

Write to the glossary file using xindy syntax.

```
4819 \glossary[\csname glo@\@gls@label @type\endcsname]{%
4820 (indexentry :tkey (\csname glo@\@gls@label @index\endcsname)

4821 :locref \string{\@glo@counterprefix}{\@glslocref}\string" %
4822 :attr \string"\@gls@counter\@glo@suffix\string"
4823 \@glo@range
4824 )
4825 }%
4826 \else
```

Convert the format information into the format required for makeindex

```
4827 \set@glo@numformat{\@glo@numfmt}{\@gls@counter}{\@glsnumberformat}%
4828 {\@glo@counterprefix}%
```

Write to the glossary file using makeindex syntax.

```
4829 \glossary[\csname glo@\@gls@label @type\endcsname]{%
4830 \string\glossaryentry{\csname glo@\@gls@label @index\endcsname
4831 \@gls@encapchar\@glo@numfmt}{\@gls@locref}}%
4832 \fi
4833 }
```

`\ls@getcounterprefix` Get the prefix that needs to be prepended to counter in order to get the hyper counter. (For example, with the standard article class and hyperref, `\theequation` needs to be prefixed with `\section num`.) to get the equivalent `\theHequation`.) NB this assumes that the prefix ends with a dot, which is the standard. (Otherwise it makes the xindy location classes more complicated.)

```
4834 \newcommand*\@gls@getcounterprefix[2]{%
4835 \edef\@gls@thisloc{#1}\edef\@gls@thisHloc{#2}%
4836 \ifx\@gls@thisloc\@gls@thisHloc
4837 \def\@glo@counterprefix{}%
4838 \else
4839 \def\@gls@get@counterprefix##1.#1##2\end@getprefix{%
4840 \def\@glo@tmp{##2}%
4841 \ifx\@glo@tmp\@empty
4842 \def\@glo@counterprefix{}%
4843 \else
4844 \def\@glo@counterprefix{##1}%
4845 \fi
4846 }%
4847 \@gls@get@counterprefix#2.#1\end@getprefix
```

Warn if no prefix can be formed.

```
4848 \ifx\@glo@counterprefix\@empty
4849 \GlossariesWarning{Hyper target ‘#2’ can’t be formed by
4850 prefixing~Jlocation ‘#1’. You need to modify the
4851 definition of \string\theH\@gls@counter~Jotherwise you
4852 will get the warning: “‘name{\@gls@counter.#1}’ has been~J
4853 referenced but does not exist”}%
4854 \fi
4855 \fi
4856 }
```

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.

```
4857 \newcommand{\@do@seeglossary}[2]{%
4858 \def\@gls@xref{#2}%
```

```

4859 \@onelevel@sanitize\@gls@xref
4860 \@gls@checkmkidxchars\@gls@xref
4861 \ifglxsindy
4862   \glossary[\csname glo@#1@type\endcsname]{%
4863     (indexentry
4864       :tkey (\csname glo@#1@index\endcsname)
4865       :xref (\string"\@gls@xref\string")
4866       :attr \string"see\string"
4867     )
4868   }%
4869 \else
4870   \glossary[\csname glo@#1@type\endcsname]{%
4871     \string\glossaryentry{\csname glo@#1@index\endcsname
4872     \@gls@encapchar glsseeformat\@gls@xref}{Z}}%
4873 \fi
4874 }

```

`\@gls@fixbraces` If no optional argument is specified, list needs to be enclosed in a set of braces.

```

4875 \def\@gls@fixbraces#1#2#3\@nil{%
4876   \ifx#2[\relax
4877     \@gls@fixbraces#1#2#3\@end@fixbraces
4878   \else
4879     \def#1{{#2#3}}%
4880   \fi
4881 }

```

`\@gls@fixbraces`

```

4882 \def\@gls@fixbraces#1[#2]#3\@end@fixbraces{%
4883   \def#1{[#2]{#3}}%
4884 }

```

`\glssee` `\glssee{<label>}{<cross-ref list>}`

```

4885 \DeclareRobustCommand*\glssee[3][\seename]{%
4886   \@do@seeglossary{#2}{[#1]{#3}}
4887 \newcommand*\@glssee[3][\seename]{%
4888   \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.

```

4889 \DeclareRobustCommand*\glsseeformat[3][\seename]{%
4890   \emph{#1} \glsseelist{#2}}

```

`\glsseelist` `\glsseelist{<list>}` formats list of entry labels.

```

4891 \DeclareRobustCommand*\glsseelist[1]{%

```

If there is only one item in the list, set the last separator to do nothing.

```

4892   \let\@gls@dolast\relax

```

Don’t display separator on the first iteration of the loop

```

4893   \let\@gls@donext\relax

```

Iterate through the labels

```
4894 \@for\@gls@thislabel:=#1\do{%
```

Check if on last iteration of loop

```
4895 \ifx\@xfor@nextelement\@nnil
4896 \@gls@dolast
4897 \else
4898 \@gls@donext
4899 \fi
```

Display the entry for this label. (Expanding label as it's a temporary control sequence that's used elsewhere.)

```
4900 \expandafter\glsseeitem\expandafter{\@gls@thislabel}%
```

Update separators

```
4901 \let\@gls@dolast\glsseelastsep
4902 \let\@gls@donext\glsseesep
4903 }%
4904 }
```

`\glsseelastsep` Separator to use between penultimate and ultimate entries in a cross-referencing list.

```
4905 \newcommand*{\glsseelastsep}{\space\andname\space}
```

`\glsseesep` Separator to use between entries in a cross-referencing list.

```
4906 \newcommand*{\glsseesep}{, }
```

`\glsseeitem` `\glsseeitem{<label>}` formats individual entry in a cross-referencing list.

```
4907 \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.)

```
4908 \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.

```
4909 \newcommand*{\gls@save@numberlist}[1]{%
4910 \ifglssavenumberlist
4911 \toks@{#1}%
4912 \edef\@do@writeaux@info{%
4913 \noexpand\csgdef{glo@\glscurrententrylabel @numberlist}{\the\toks@}%
```

```

4914    }%
4915    \@onelevel@sanitize\@do@writeaux@info
4916    \protected@write\@auxout{}\@do@writeaux@info}%
4917    \fi
4918 }

```

`\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.)

```

4919 \newcommand*\warn@noprintglossary{}\%

```

`\printglossary` The TOC title needs to be processed in a different manner to the main title in case the translator and hyperref packages are both being used.

```

4920 \ifcsundef{printglossary}\%
4921 {%

```

If `\printglossary` is already defined, issue a warning and undefine it.

```

4922   \@gls@warnonglossdefined
4923   \undef\printglossary
4924 }

```

`\printglossary` has an optional argument. The default value is to set the glossary type to the main glossary.

```

4925 \newcommand*\printglossary[1][type=\glsdefaulttype]{%
4926   \@printglossary{#1}\@print@glossary}%
4927 }

```

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`

```

4928 \newcommand*\printglossaries{%
4929   \forallglossaries{\@glo@type}\printglossary[type=\@glo@type]}%
4930 }

```

`\printnoidxglossary` Provide an alternative to `\printglossary` that doesn't require an external indexing application. Entries won't be sorted and the location list will be empty.

```

4931 \newcommand*\printnoidxglossary[1][type=\glsdefaulttype]{%
4932   \@printglossary{#1}\@print@noidx@glossary}%
4933 }

```


`\printnoidxglossaries` Analogous to `\printglossaries`

```

4934 \newcommand*{\printnoidxglossaries}{%
4935   \forallglossaries{\@glo@type}{\printnoidxglossary[type=\@glo@type]}%
4936 }

```

`\@printgloss@setsort` Initialise to do nothing.

```

4937 \newcommand*{\@printgloss@setsort}{}

```

`\@printglossary` Sets up the glossary for either `\printglossary` or `\printnoidxglossary`. The first argument is the options list, the second argument is the handler macro that deals with the actual glossary.

```

4938 \newcommand{\@printglossary}[2]{%

```

Set up defaults.

```

4939   \def\@glo@type{\glsdefaulttype}%
4940   \def\glossarytitle{\csname @glo@type\@glo@type @title\endcsname}%

```

```

4941   \def\glossarytoctitle{\glossarytitle}%
4942   \let\org@glossarytitle\glossarytitle
4943   \def\@glossarystyle{}%
4944   \def\gls@dotocitle{\glssettocitle{\@glo@type}}%

```

Store current value of `\glossaryentrynumbers`. (This may be changed via the optional argument)

```

4945   \let\@org@glossaryentrynumbers\glossaryentrynumbers

```

Localise the effects of the optional argument

```

4946   \bgroup

```

Activate or deactivate sort key:

```

4947   \@printgloss@setsort

```

Determine settings specified in the optional argument.

```

4948   \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)

```

4949   \ifx\glossarytitle\org@glossarytitle
4950   \else
4951     \expandafter\let\csname @glo@type\@glo@type @title\endcsname
4952       \glossarytitle
4953   \fi

```

Allow a high-level user command to indicate the current glossary

```

4954   \let\currentglossary\@glo@type

```

Enable individual number lists to be suppressed.

```

4955   \let\org@glossaryentrynumbers\glossaryentrynumbers
4956   \let\glsnonextpages\@glsnonextpages

```

Enable individual number list to be activated:

```

4957   \let\glsnextpages\@glsnextpages

```

Enable suppression of description terminators.

```
4958 \let\nopostdesc\@nopostdesc
```

Set up the entry for the TOC

```
4959 \gls@dotoc@title
```

Set the glossary style

```
4960 \@glossarystyle
```

Added a way to fetch the current entry label (v3.08 updated for new `\glossentry` and `\subglossentry`, but this is now only needed for backward compatibility):

```
4961 \let\gls@org@glossaryentryfield\glossentry
4962 \let\gls@org@glossarysubentryfield\subglossentry
4963 \renewcommand{\glossentry}[1]{%
4964   \xdef\glscurrententrylabel{\glsdetoklabel{##1}}%
4965   \gls@org@glossaryentryfield{##1}%
4966 }%
4967 \renewcommand{\subglossentry}[2]{%
4968   \xdef\glscurrententrylabel{\glsdetoklabel{##2}}%
4969   \gls@org@glossarysubentryfield{##1}{##2}%
4970 }%
```

Now do the handler macro that deals with the actual glossary:

```
4971 #2%
```

End the current scope

```
4972 \egroup
```

Reset `\glossaryentrynumbers`

```
4973 \global\let\glossaryentrynumbers\@org@glossaryentrynumbers
```

Suppress warning about no `\printglossary`

```
4974 \global\let\warn@noprintglossary\relax
4975 }
```

`\@print@glossary` Internal workings of `\printglossary` dealing with reading the external file.

```
4976 \newcommand{\@print@glossary}{%
```

Some macros may end up being expanded into internals in the glossary, so need to make `@` a letter. (Unlikely to be a problem since v3.08a but kept for backward compatibility.)

```
4977 \makeatletter
```

Input the glossary file, if it exists.

```
4978 \@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.

```
4979 \IfFileExists{\jobname.\csname @glo@type\@glo@type @in\endcsname}%
4980 {}%
4981 {\null}%
```

If xindy is being used, need to write the language dependent information to the .aux file for makeglossaries.

```

4982 \ifglxindy
4983 \ifcsundef{@xdy@\@glo@type @language}%
4984 {%
4985 \edef\@do@auxoutstuff{%
4986 \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.

```

4987 \noexpand\immediate\noexpand\write\@auxout{%
4988 \string\providecommand\string\@xdylanguage[2]{}%
4989 \noexpand\immediate\noexpand\write\@auxout{%
4990 \string\@xdylanguage{\@glo@type}{\@xdy@main@language}}%
4991 }%
4992 }%
4993 }%
4994 {%
4995 \edef\@do@auxoutstuff{%
4996 \noexpand\AtEndDocument{%
4997 \noexpand\immediate\noexpand\write\@auxout{%
4998 \string\providecommand\string\@xdylanguage[2]{}%
4999 \noexpand\immediate\noexpand\write\@auxout{%
5000 \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type
5001 @language\endcsname}}%
5002 }%
5003 }%
5004 }%
5005 \@do@auxoutstuff
5006 \edef\@do@auxoutstuff{%
5007 \noexpand\AtEndDocument{%

```

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

```

5008 \noexpand\immediate\noexpand\write\@auxout{%
5009 \string\providecommand\string\@gls@codepage[2]{}%
5010 \noexpand\immediate\noexpand\write\@auxout{%
5011 \string\@gls@codepage{\@glo@type}{\gls@codepage}}%
5012 }%
5013 }%
5014 \@do@auxoutstuff
5015 \fi

```

Activate warning if \makeglossaries hasn't been used.

```

5016 \renewcommand*{\@warn@nomakeglossaries}{%
5017 \GlossariesWarningNoLine{\string\makeglossaries\space
5018 hasn't been used,^^Jthe glossaries will not be updated}%
5019 }%

```

5020 }

The sort macros all have the syntax:

`\@glo@sortmacro@<order>{<type>}`

where *<order>* is the sort order as specified by the sort key and *<type>* is the glossary type. (The referenced entry list is stored in `\@glsref@<type>`). The actual sorting is done by `\@glo@sortentries{<handler>}{<type>}`.

`\@glo@sortentries`

```
5021 \newcommand*{\@glo@sortentries}[2]{%
5022   \def\@glo@sortinglist{}%
5023   \def\@glo@sortinghandler{#1}%
5024   \edef\@glo@type{#2}%
5025   \forlistcsloop{\@glo@do@sortentries}{\@glsref@#2}%
5026   \csdef{\@glsref@#2}{}%
5027   \@for\@this@label:=\@glo@sortinglist\do{%
```

Has this entry already been added?

```
5028     \xifinlistcs{\@this@label}{\@glsref@#2}%
5029     {}%
5030     {%
5031       \listcsxadd{\@glsref@#2}{\@this@label}%
5032     }%
5033     \ifcsdef{\@glo@sortingchildren@ \@this@label}%
5034     {%
5035       \@glo@addchildren{#2}{\@this@label}%
5036     }%
5037     {}%
5038   }%
5039 }
```

`\@glo@addchildren`

`\@glo@addchildren{<type>}{<parent>}`

```
5040 \newcommand*{\@glo@addchildren}[2]{%
```

Scope to allow nesting.

```
5041   \bgroup
5042   \letcs{\@glo@childlist}{\@glo@sortingchildren@#2}%
5043   \@for\@this@childlabel:=\@glo@childlist\do
5044   {%
```

Check this label hasn't already been added.

```
5045     \xifinlistcs{\@this@childlabel}{\@glsref@#1}%
5046     {}%
5047     {%
5048       \listcsxadd{\@glsref@#1}{\@this@childlabel}%
5049     }%
```

Does this child have children?

```

5050      \ifcsdef{@glo@sortingchildren@ \@this@childlabel}%
5051      {%
5052      \@glo@addchildren{#1}{\@this@childlabel}%
5053      }%
5054      {%
5055      }%
5056      }%
5057  \egroup
5058 }

```

@glo@do@sortentries

```

5059 \newcommand*{\@glo@do@sortentries}[1]{%
5060   \ifglshasparent{#1}%
5061   {%

```

This entry has a parent, so add it to the child list

```

5062   \edef\@glo@parent{\csuse{glo@glsetoklabel{#1}@parent}}%
5063   \ifcsundef{@glo@sortingchildren@\@glo@parent}%
5064   {%
5065     \csdef{@glo@sortingchildren@\@glo@parent}{}%
5066   }%
5067   {}%
5068   \expandafter\@glo@sortedinsert
5069     \csname @glo@sortingchildren@\@glo@parent\endcsname{#1}%

```

Has the parent been added?

```

5070   \xifinlistcs{\@glo@parent}{\glstrref@\@glo@type}%
5071   {%

```

Yes, it has so do nothing.

```

5072   }%
5073   {%

```

No, it hasn't so add it now.

```

5074     \expandafter\@glo@do@sortentries\expandafter{\@glo@parent}%
5075     }%
5076   }%
5077   {%
5078     \@glo@sortedinsert{\@glo@sortinglist}{#1}%
5079   }%
5080 }

```

\@glo@sortedinsert `\@glo@sortedinsert{<list>}{<entry label>}`

Insert into list.

```

5081 \newcommand*{\@glo@sortedinsert}[2]{%
5082   \dtlinsertinto{#2}{#1}{\@glo@sortinghandler}%
5083 }%

```

The sort handlers need to be in the form required by datatool's `\dtl@sortlist` macro. These must set the count register `\dtl@sortresult` to either -1 ($\#1$ less than $\#2$), 0 ($\#1 = \#2$) or $+1$ ($\#1$ greater than $\#2$).

`\lo@sorthandler@word`

```

5084 \newcommand*{\@glo@sorthandler@word}[2]{%
5085   \letcs\@gls@sort@A{glo\glsdetoklabel{\#1}@sort}%
5086   \letcs\@gls@sort@B{glo\glsdetoklabel{\#2}@sort}%
5087   \edef\glo@do@compare{%
5088     \noexpand\dtlwordindexcompare{\noexpand\dtl@sortresult}%
5089     {\expandonce\@gls@sort@B}%
5090     {\expandonce\@gls@sort@A}%
5091   }%
5092   \glo@do@compare
5093 }
```

`@sorthandler@letter`

```

5094 \newcommand*{\@glo@sorthandler@letter}[2]{%
5095   \letcs\@gls@sort@A{glo\glsdetoklabel{\#1}@sort}%
5096   \letcs\@gls@sort@B{glo\glsdetoklabel{\#2}@sort}%
5097   \edef\glo@do@compare{%
5098     \noexpand\dtlletterindexcompare{\noexpand\dtl@sortresult}%
5099     {\expandonce\@gls@sort@B}%
5100     {\expandonce\@gls@sort@A}%
5101   }%
5102   \glo@do@compare
5103 }
```

`\lo@sorthandler@case` Case-sensitive sort.

```

5104 \newcommand*{\@glo@sorthandler@case}[2]{%
5105   \letcs\@gls@sort@A{glo\glsdetoklabel{\#1}@sort}%
5106   \letcs\@gls@sort@B{glo\glsdetoklabel{\#2}@sort}%
5107   \edef\glo@do@compare{%
5108     \noexpand\dtlcompare{\noexpand\dtl@sortresult}%
5109     {\expandonce\@gls@sort@B}%
5110     {\expandonce\@gls@sort@A}%
5111   }%
5112   \glo@do@compare
5113 }
```

`@sorthandler@nocase` Case-insensitive sort.

```

5114 \newcommand*{\@glo@sorthandler@nocase}[2]{%
5115   \letcs\@gls@sort@A{glo\glsdetoklabel{\#1}@sort}%
5116   \letcs\@gls@sort@B{glo\glsdetoklabel{\#2}@sort}%
5117   \edef\glo@do@compare{%
5118     \noexpand\dtlicompare{\noexpand\dtl@sortresult}%
5119     {\expandonce\@gls@sort@B}%
5120     {\expandonce\@gls@sort@A}%
5121   }%
```

```

5122 \glo@do@compare
5123 }

@glo@sortmacro@word  Sort macro for ‘word’

5124 \newcommand*{\@glo@sortmacro@word}[1]{%
5125 \ifdefstring{\@glo@default@sorttype}{standard}%
5126 {%
5127 \@glo@sortentries{\@glo@sorthandler@word}{#1}%
5128 }%
5129 {%
5130 \PackageError{glossaries}{Conflicting sort options:^^J
5131 \string\usepackage[sort=\@glo@default@sorttype]{glossaries}^^J
5132 \string\printnoidxglossary[sort=word]}{}%
5133 }%
5134 }

@lo@sortmacro@letter  Sort macro for ‘letter’

5135 \newcommand*{\@glo@sortmacro@letter}[1]{%
5136 \ifdefstring{\@glo@default@sorttype}{standard}%
5137 {%
5138 \@glo@sortentries{\@glo@sorthandler@letter}{#1}%
5139 }%
5140 {%
5141 \PackageError{glossaries}{Conflicting sort options:^^J
5142 \string\usepackage[sort=\@glo@default@sorttype]{glossaries}^^J
5143 \string\printnoidxglossary[sort=letter]}{}%
5144 }%
5145 }

@sortmacro@standard  Sort macro for ‘standard’. (Use either ‘word’ or ‘letter’ order.)

5146 \newcommand*{\@glo@sortmacro@standard}[1]{%
5147 \ifdefstring{\@glo@default@sorttype}{standard}%
5148 {%
5149 \ifcsdef{\@glo@sorthandler@\glsorder}%
5150 {%
5151 \@glo@sortentries{\csuse{\@glo@sorthandler@\glsorder}}{#1}%
5152 }%
5153 {%
5154 \PackageError{glossaries}{Unknown sort handler ‘\glsorder’}{}%
5155 }%
5156 }%
5157 {%
5158 \PackageError{glossaries}{Conflicting sort options:^^J
5159 \string\usepackage[sort=\@glo@default@sorttype]{glossaries}^^J
5160 \string\printnoidxglossary[sort=standard]}{}%
5161 }%
5162 }

@glo@sortmacro@case  Sort macro for ‘case’

```

```

5163 \newcommand*{\@glo@sortmacro@case}[1]{%
5164   \ifdefstring{\@glo@default@sorttype}{standard}%
5165   {%
5166     \@glo@sortentries{\@glo@sorthandler@case}{#1}%
5167   }%
5168   {%
5169     \PackageError{glossaries}{Conflicting sort options:^^J
5170       \string\usepackage[sort=\@glo@default@sorttype]{glossaries}^^J
5171       \string\printnoidxglossary[sort=case]}{}%
5172   }%
5173 }

```

`\lo@sortmacro@nocase` Sort macro for ‘nocase’

```

5174 \newcommand*{\@glo@sortmacro@nocase}[1]{%
5175   \ifdefstring{\@glo@default@sorttype}{standard}%
5176   {%
5177     \@glo@sortentries{\@glo@sorthandler@nocase}{#1}%
5178   }%
5179   {%
5180     \PackageError{glossaries}{Conflicting sort options:^^J
5181       \string\usepackage[sort=\@glo@default@sorttype]{glossaries}^^J
5182       \string\printnoidxglossary[sort=nocase]}{}%
5183   }%
5184 }

```

`\@glo@sortmacro@def` Sort macro for ‘def’. The order of definition is given in `\glolist@⟨type⟩`.

```

5185 \newcommand*{\@glo@sortmacro@def}[1]{%
5186   \def\@glo@sortinglist{}%
5187   \forlslentries[#1]{\@gls@thislabel}%
5188   {%
5189     \xifinlistcs{\@gls@thislabel}{\@glsref@#1}%
5190     {%
5191       \listeadadd{\@glo@sortinglist}{\@gls@thislabel}%
5192     }%
5193   }%

```

Hasn’t been referenced.

```

5194   }%
5195 }%
5196 \cslet{\@glsref@#1}{\@glo@sortinglist}%
5197 }

```

`\lo@sortmacro@def@do` This won’t include parent entries that haven’t been referenced.

```

5198 \newcommand*{\@glo@sortmacro@def@do}[1]{%
5199   \ifinlistcs{#1}{\@glsref@\@glo@type}%
5200   {}%
5201   {%
5202     \listcsadd{\@glsref@\@glo@type}{#1}%
5203   }%

```



```

5204 \ifcsdef{@glo@sortingchildren@#1}%
5205 {%
5206   \@glo@addchildren{\@glo@type}{#1}%
5207 }%
5208 {}%
5209 }

\@glo@sortmacro@use Sort macro for 'use'. (No sorting is required, as the entries are already in order
of use, so do nothing.)
5210 \newcommand*{\@glo@sortmacro@use}[1]{%

\print@noidx@glossary Glossary handler for \printnoidxglossary which doesn't use an indexing ap-
plication. Since \printnoidxglossary may occur at the start of the docu-
ment, we can't just check if an entry has been used. Instead, the first pass needs
to write information to the aux file every time an entry is referenced. This needs
to be read in on the second run and stored in a list corresponding to the appro-
priate glossary.
5211 \newcommand*{\@print@noidx@glossary}{%
5212   \ifcsdef{@glsref@ \@glo@type}%
5213   {%
Sort the entries:
5214     \ifcsdef{@glo@sortmacro@ \@glo@sorttype}%
5215     {%
5216       \csuse{@glo@sortmacro@ \@glo@sorttype}{\@glo@type}%
5217     }%
5218     {%
5219       \PackageError{glossaries}{Unknown sort handler '@@glo@sorttype'}{}%
5220     }%
Do the glossary heading and preamble
5221     \glossarysection[\glossarytoctitle]{\glossarytitle}%
5222     \glossarypreamble
5223     \begin{theglossary}%
5224     \glossaryheader
5225     \glsresetentrylist
5226     \def\@gls@currentlettergroup{}%
Iterate through the entries.
5227     \forlistcsloop{\@gls@noidx@do}{\@glsref@ \@glo@type}%
Finally end the glossary and do the postamble:
5228     \end{theglossary}%
5229     \glossarypostamble
5230   }%
5231   {%
5232     \@gls@noref@warn{\@glo@type}%
5233   }%
5234 }

```

\glo@grabfirst

```
5235 \def\glo@grabfirst#1#2\@nil{%
5236   \def\@gls@firsttok{#1}%
5237   \ifdefempty\@gls@firsttok
5238     {%
5239       \def\@glo@thislettergrp{0}%
5240     }%
5241     {%
      Sanitize it:
5242       \@onelevel@sanitize\@gls@firsttok
      Fetch the first letter:
5243       \expandafter\@glo@grabfirst\@gls@firsttok{}{}\@nil
5244     }%
5245 }
```

\@glo@grabfirst

```
5246 \def\@glo@grabfirst#1#2\@nil{%
5247   \ifdefempty\@glo@thislettergrp
5248     {%
5249       \def\@glo@thislettergrp{glssymbols}%
5250     }%
5251     {%
5252       \count@=\uccode'#1\relax
5253       \ifnum\count@=0\relax
5254         \def\@glo@thislettergrp{glssymbols}%
5255       \else
5256         \ifdefstring\@glo@sorttype{case}%
5257         {%
5258           \count@='#1\relax
5259         }%
5260         {%
5261         }%
5262         \edef\@glo@thislettergrp{\the\count@}%
5263       \fi
5264     }%
5265 }
```

\@gls@noidx@do Handler for list iteration used by \@print@noidx@glossary. The argument is the entry label. This only allows one sublevel.

```
5266 \newcommand{\@gls@noidx@do}[1]{%
      Get this entry's location list
5267   \letcs{\@gls@loclist}{\glo@glstdetoklabel{#1}@loclist}%
      Does this entry have a parent?
5268   \ifglshasparent{#1}%
5269   {%
```

Has a parent.

```
5270 \gls@level=\csuse{glo@glsdetoklabel{#1}@level}\relax
5271 \ifdefvoid{\@gls@loclist}
5272 {%
5273   \subglossentry{\gls@level}{#1}{}%
5274 }%
5275 {%
5276   \subglossentry{\gls@level}{#1}%
5277   {%
5278     \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
5279   }%
5280 }%
5281 }%
5282 {%
```

Doesn't have a parent Get this entry's sort key

```
5283 \letcs{\@gls@sort}{glo@glsdetoklabel{#1}@sort}%
```

Fetch the first letter:

```
5284 \expandafter\glo@grabfirst\@gls@sort{}{}\@nil
5285 \ifdefequal{\@glo@thislettergrp}{\@gls@currentlettergroup}%
5286 {}%
5287 {%
```

Do the group header:

```
5288 \ifdefempty{\@gls@currentlettergroup}{\@gls@groupskip}%
5289 \gls@groupheading{\@glo@thislettergrp}%
5290 }%
5291 \let\@gls@currentlettergroup\@glo@thislettergrp
```

Do this entry:

```
5292 \ifdefvoid{\@gls@loclist}
5293 {%
5294   \glossentry{#1}{}%
5295 }%
5296 {%
5297   \glossentry{#1}%
5298   {%
5299     \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
5300   }%
5301 }%
5302 }%
5303 }
```

`\glsnoidxloclist` `\glsnoidxloclist{<list cs>}`

Display location list.

```
5304 \newcommand*{\glsnoidxloclist}[1]{%
5305   \def\@gls@noidxloclist@sep{%
```

```

5306 \def\@gls@noidxloclist@prev{}%
5307 \forlistloop{\glsnoidxloclisthandler}{#1}%
5308 }

```

`noidxloclisthandler` Handler for location list iterator.

```

5309 \newcommand*\glsnoidxloclisthandler[1]{%
5310 \ifdefstring{\@gls@noidxloclist@prev}{#1}%
5311 {%
    Same as previous location so skip.
5312 }%
5313 {%
5314 \@gls@noidxloclist@sep
5315 #1%
5316 \def\@gls@noidxloclist@sep{\delimN}%
5317 \def\@gls@noidxloclist@prev{#1}%
5318 }%
5319 }

```

`splayloclisthandler` Handler for location list iterator when used with `\glsdisplaynumberlist`.

```

5320 \newcommand*\glsnoidxdisplayloclisthandler[1]{%
5321 \ifdefstring{\@gls@noidxloclist@prev}{#1}%
5322 {%
    Same as previous location so skip.
5323 }%
5324 {%
5325 \@gls@noidxloclist@sep
5326 \@gls@noidxloclist@prev
5327 \def\@gls@noidxloclist@prev{#1}%
5328 }%
5329 }

```

`\glsnoidxdisplayloc` `\glsnoidxdisplayloc{<prefix>}{<counter>}{<format>}{<location>}`

Display a location in the location list.

```

5330 \newcommand*\glsnoidxdisplayloc[4]{%
5331 \setentrycounter[#1]{#2}%
5332 \csuse{#3}{#4}%
5333 }

```

`\@gls@reference` `\@gls@reference{<type>}{<label>}{<loc>}`

Identifies that a reference has been used (for use in the aux file). All entries must be defined in the preamble.

```

5334 \newcommand*\@gls@reference[3]{%

```

Add to label list

```

5335 \glsdoifexistsorwarn{#2}%
5336 {%
5337   \ifcsundef{@glsref@#1}{\csgdef{@glsref@#1}{}}{}%
5338   \ifinlistcs{#2}{@glsref@#1}%
5339   {}%
5340   {\listcsgadd{@glsref@#1}{#2}}%

```

Add to location list

```

5341 \ifcsundef{glo@glsdetoklabel{#2}@loclist}%
5342 {\csgdef{glo@glsdetoklabel{#2}@loclist}{}}%
5343 {}%
5344 \listcsgadd{glo@glsdetoklabel{#2}@loclist}{#3}%
5345 }%
5346 }

```

The keys that can be used in the optional argument to `\printglossary` or `\printnoidxglossary` are as follows: The `type` key sets the glossary type.

```

5347 \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`.

```

5348 \define@key{printgloss}{title}{%
5349 \def@glossarytitle{#1}%
5350 \let@gls@dotoc@title\relax
5351 }

```

The `toctitle` sets the text used for the relevant entry in the table of contents.

```

5352 \define@key{printgloss}{toctitle}{%
5353 \def@glossarytoctitle{#1}%
5354 \let@gls@dotoc@title\relax
5355 }

```

The `style` key sets the glossary style (but only for the given glossary).

```

5356 \define@key{printgloss}{style}{%
5357 \ifcsundef{@glsstyle@#1}%
5358 {%
5359   \PackageError{glossaries}%
5360   {Glossary style ‘#1’ undefined}{}%
5361 }%
5362 {%
5363   \def@glossarystyle{\setglossentrycompatibility
5364     \csname @glsstyle@#1\endcsname}%
5365 }%
5366 }

```

The `numberedsection` key determines if this glossary should be in a numbered section.

```

5367 \define@choicekey{printgloss}{numberedsection}[\val\nr]{%
5368 false,nolabel,autolabel,nameref}[nolabel]{%
5369 \ifcase\nr\relax

```

```

5370 \renewcommand*{\@@glossarysecstar}{*}%
5371 \renewcommand*{\@@glossaryseclabel}{}%
5372 \or
5373 \renewcommand*{\@@glossarysecstar}{}%
5374 \renewcommand*{\@@glossaryseclabel}{}%
5375 \or
5376 \renewcommand*{\@@glossarysecstar}{}%
5377 \renewcommand*{\@@glossaryseclabel}{\label{\glsautoprefix\@glo@type}}%
5378 \or
5379 \renewcommand*{\@@glossarysecstar}{*}%
5380 \renewcommand*{\@@glossaryseclabel}{%
5381 \protected@edef\@currentlabelname{\glossarytoctitle}%
5382 \label{\glsautoprefix\@glo@type}}%
5383 \fi
5384 }

```

The `nogroupskip` key determines whether or not there should be a vertical gap between glossary groups.

```

5385 \define@choicekey{printgloss}{nogroupskip}{true,false}[true]{%
5386 \csuse{glsnogroupskip#1}%
5387 }

```

The `nonumberlist` key determines if this glossary should have a number list.

```

5388 \define@boolkey{printgloss}[gls]{nonumberlist}[true]{%
5389 \ifglslnonumberlist
5390 \def\glossaryentrynumbers##1{}%
5391 \else
5392 \def\glossaryentrynumbers##1{##1}%
5393 \fi}

```

The `sort` key sets the glossary sort handler (`\printnoidxglossary` only).

```

5394 \define@key{printgloss}{sort}{\@glo@assign@sortkey{#1}}

```

`\@no@assign@sortkey` Issue error if used with `\printglossary`

```

5395 \newcommand*{\@glo@no@assign@sortkey}[1]{%
5396 \PackageError{glossaries}{‘sort’ key not permitted with
5397 \string\printglossary}%
5398 {The ‘sort’ key may only be used with \string\printnoidxglossary}%
5399 }

```

`\@glo@assign@sortkey` For use with `\printnoidxglossary`

```

5400 \newcommand*{\@glo@assign@sortkey}[1]{%
5401 \def\@glo@sorttype{#1}%
5402 }

```

`\@glslnonextpages` 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 `\glslnonextpages` is placed 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.

```

5403 \newcommand*{\@glsnonextpages}{%
5404   \gdef\glossaryentrynumbers##1{%
5405     \glsresetentrylist
5406   }%
5407 }

```

`\@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 placed 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.

```

5408 \newcommand*{\@glsnextpages}{%
5409   \gdef\glossaryentrynumbers##1{%
5410     ##1\glsresetentrylist}}

```

`\glsresetentrylist` Resets `\glossaryentrynumbers`

```

5411 \newcommand*{\glsresetentrylist}{%
5412   \global\let\glossaryentrynumbers\org@glossaryentrynumbers}

```

`\glsnonextpages` Outside of `\printglossary` this does nothing.

```

5413 \newcommand*{\glsnonextpages}{}

```

`\glsnextpages` Outside of `\printglossary` this does nothing.

```

5414 \newcommand*{\glsnextpages}{}

```

`glossaryentry` If the `entrycounter` package option has been used, define a counter to number each level 0 entry.

```

5415 \ifglentrycounter
5416   \ifx\@gls@counterwithin\@empty
5417     \newcounter{glossaryentry}
5418   \else
5419     \newcounter{glossaryentry}[\@gls@counterwithin]
5420   \fi
5421   \def\theHglossaryentry{\currentglossary.\theglossaryentry}
5422 \fi

```

`glossarysubentry` If the `subentrycounter` package option has been used, define a counter to number each level 1 entry.

```

5423 \ifglssubentrycounter
5424   \ifglentrycounter
5425     \newcounter{glossarysubentry}[glossaryentry]
5426   \else
5427     \newcounter{glossarysubentry}
5428   \fi
5429   \def\theHglossarysubentry{\currentglssubentry.\theglossarysubentry}
5430 \fi

```

`\resetsubentrycounter` Resets the glossarysubentry counter.

```
5431 \ifglssubentrycounter
5432   \newcommand*{\glsresetsubentrycounter}{%
5433     \setcounter{glossarysubentry}{0}%
5434   }
5435 \else
5436   \newcommand*{\glsresetsubentrycounter}{}
5437 \fi
```

`\resetsubentrycounter` Resets the glossareentry counter.

```
5438 \ifglseentrycounter
5439   \newcommand*{\glsresetentrycounter}{%
5440     \setcounter{glossaryentry}{0}%
5441   }
5442 \else
5443   \newcommand*{\glsresetentrycounter}{}
5444 \fi
```

`\glsstepentry` Advance the glossaryentry counter if in use. The argument is the label associated with the entry.

```
5445 \ifglseentrycounter
5446   \newcommand*{\glsstepentry}[1]{%
5447     \refstepcounter{glossaryentry}%
5448     \label{glsentry-\glsdetoklabel{#1}}%
5449   }
5450 \else
5451   \newcommand*{\glsstepentry}[1]{}
5452 \fi
```

`\glsstepsubentry` Advance the glossarysubentry counter if in use. The argument is the label associated with the subentry.

```
5453 \ifglssubentrycounter
5454   \newcommand*{\glsstepsubentry}[1]{%
5455     \edef\currentglssubentry{\glsdetoklabel{#1}}%
5456     \refstepcounter{glossarysubentry}%
5457     \label{glsentry-\currentglssubentry}%
5458   }
5459 \else
5460   \newcommand*{\glsstepsubentry}[1]{}
5461 \fi
```

`\glsrefentry` Reference the entry or sub-entry counter if in use, otherwise just do `\gls`.

```
5462 \ifglseentrycounter
5463   \newcommand*{\glsrefentry}[1]{\ref{glsentry-\glsdetoklabel{#1}}}
5464 \else
5465   \ifglssubentrycounter
5466     \newcommand*{\glsrefentry}[1]{\ref{glsentry-\glsdetoklabel{#1}}}
5467   \else
```



```

5468 \newcommand*{\glsrefentry}[1]{\gls{#1}}
5469 \fi
5470 \fi

```

`\glsentrycounterlabel` Defines how to display the glossaryentry counter.

```

5471 \ifglsentrycounter
5472 \newcommand*{\glsentrycounterlabel}{\theglossaryentry.\space}
5473 \else
5474 \newcommand*{\glsentrycounterlabel}{}
5475 \fi

```

`\glsentrysubentrycounterlabel` Defines how to display the glossarysubentry counter.

```

5476 \ifglssubentrycounter
5477 \newcommand*{\glsentrysubentrycounterlabel}{\theglossarysubentry)\space}
5478 \else
5479 \newcommand*{\glsentrysubentrycounterlabel}{}
5480 \fi

```

`\glsentryitem` Step and display glossaryentry counter, if appropriate.

```

5481 \ifglsentrycounter
5482 \newcommand*{\glsentryitem}[1]{%
5483 \glsstepentry{#1}\glsentrycounterlabel
5484 }
5485 \else
5486 \newcommand*{\glsentryitem}[1]{\glsresetsubentrycounter}
5487 \fi

```

`\glssubentryitem` Step and display glossarysubentry counter, if appropriate.

```

5488 \ifglssubentrycounter
5489 \newcommand*{\glssubentryitem}[1]{%
5490 \glsstepsubentry{#1}\glssubentrycounterlabel
5491 }
5492 \else
5493 \newcommand*{\glssubentryitem}[1]{}
5494 \fi

```

`\theglossary` If the `\theglossary` environment has already been defined, a warning will be issued. This environment should be redefined by glossary styles.

```

5495 \ifcsundef{theglossary}%
5496 {%
5497 \newenvironment{theglossary}{}{}}%
5498 }%
5499 {%
5500 \@gls@warnontheGLOSSdefined
5501 \renewenvironment{theglossary}{}{}}%
5502 }

```

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

```
5503 \newcommand*{\glossaryheader}{}

```

\glstarget \glstarget{<label>}{<name>}

Provide user interface to \@glstarget to make it easier to modify the glossary style in the document.

```
5504 \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>}

```
5505 \providecommand*{\compatibleglossentry}[2]{%
5506   \toks@{#2}%
5507   \protected@edef\@do@glossentry{\noexpand\glossaryentryfield{#1}%
5508     {\noexpand\glsnamefont
5509       {\expandafter\expandonce\csname glo@#1@name\endcsname}}}%
5510     {\expandafter\expandonce\csname glo@#1@desc\endcsname}%
5511     {\expandafter\expandonce\csname glo@#1@symbol\endcsname}%
5512     {\the\toks@}}%
5513   }%
5514   \@do@glossentry
5515 }
```

\glossentryname

```
5516 \newcommand*{\glossentryname}[1]{%
5517   \glsdoifexistsorwarn{#1}%
5518   {%
5519     \letcs{\glo@name}{\glo@\glsdetoklabel{#1}@name}%
5520     \expandafter\glsnamefont\expandafter{\glo@name}%
5521   }%
5522 }
```

\Glossentryname

```
5523 \newcommand*{\Glossentryname}[1]{%
5524   \glsdoifexistsorwarn{#1}%
5525   {%
5526     \glsnamefont{\Glsentryname{#1}}%

```

```

5527 }%
5528 }

```

`\glossentrydesc`

```

5529 \newcommand*{\glossentrydesc}[1]{%
5530   \glsdoifexistsorwarn{#1}%
5531   {%
5532     \glsentrydesc{#1}%
5533   }%
5534 }

```

`\Glossentrydesc`

```

5535 \newcommand*{\Glossentrydesc}[1]{%
5536   \glsdoifexistsorwarn{#1}%
5537   {%
5538     \Glsentrydesc{#1}%
5539   }%
5540 }

```

`\glossentrysymbol`

```

5541 \newcommand*{\glossentrysymbol}[1]{%
5542   \glsdoifexistsorwarn{#1}%
5543   {%
5544     \glsentrysymbol{#1}%
5545   }%
5546 }

```

`\Glossentrysymbol`

```

5547 \newcommand*{\Glossentrysymbol}[1]{%
5548   \glsdoifexistsorwarn{#1}%
5549   {%
5550     \Glsentrysymbol{#1}%
5551   }%
5552 }

```

`compatiblesubglossentry` `\subglossentry{<level>}{<label>}{<page-list>}`

```

5553 \providecommand*{\compatiblesubglossentry}[3]{%
5554   \toks@{#3}%
5555   \protected@edef\@do@subglossentry{\noexpand\glossarysubentryfield{\number#1}%
5556     {#2}%
5557     {\noexpand\glsnamefont
5558       {\expandafter\expandonce\csname glo@#2@name\endcsname}}}%
5559   {\expandafter\expandonce\csname glo@#2@desc\endcsname}%
5560   {\expandafter\expandonce\csname glo@#2@symbol\endcsname}%
5561   {\the\toks@}%
5562 }%

```

```

5563 \@do@subglossentry
5564 }

```

sentrycompatibility

```

5565 \newcommand*{\setglossentrycompatibility}{%
5566   \let\glossentry\compatibleglossentry
5567   \let\subglossentry\compatiblesubglossentry
5568 }
5569 \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.

```

5570 \newcommand{\glossaryentryfield}[5]{%
5571   \GlossariesWarning
5572   {Deprecated use of \string\glossaryentryfield.^^J
5573     I recommend you change to \string\glossentry.^^J
5574     If you've just upgraded, try removing your gls auxiliary
5575     files^^J and recompile}%
5576   \noindent\textbf{\glstarget{#1}{#2}} #4 #3. #5\par}

```

lossarysubentryfield

```
\glossarysubentryfield{<level>}{<label>}{<name>}{<description>}{<symbol>}{<page-list>}
```

This command governs how each subentry should be formatted in the glossary. Glossary styles need to redefine this command. Most of the predefined styles ignore *<symbol>*. The first argument is a number indicating the level. (The level should be greater than or equal to 1.)

```

5577 \newcommand*{\glossarysubentryfield}[6]{%
5578   \GlossariesWarning
5579   {Deprecated use of \string\glossarysubentryfield.^^J
5580     I recommend you change to \string\subglossentry.^^J
5581     If you've just upgraded, try removing your gls auxiliary
5582     files^^J and recompile}%
5583   \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`

```
5584 \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`

```
5585 \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`

```
5586 \newcommand*{\glsgetgrouptitle}[1]{%
5587   \@gls@getgrouptitle{#1}{\@gls@grptitle}%
5588   \@gls@grptitle
5589 }
```

`\@gls@getgrouptitle` Gets the group title specified by the label (first argument) and stores in the second argument, which must be a control sequence.

```
5590 \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.

```

5591 \dtl@ifsingle{#1}%
5592 {%
5593   \ifcsundef{#1groupname}{\def#2{#1}}{\letcs#2{#1groupname}}%
5594 }%
5595 {%
5596   \ifboolexpr{test{\ifstrequal{#1}{glssymbols}}
5597               or test{\ifstrequal{#1}{glsnumbers}}}%
5598   {%
5599     \ifcsundef{#1groupname}{\def#2{#1}}{\letcs#2{#1groupname}}%
5600   }%
5601   {%
5602     \def#2{#1}%
5603   }%
5604 }%
5605 }

```

`@getothergrouptitle` Version for the no-indexing app option:

```

5606 \newcommand*{\@gls@noidx@getgrouptitle}[2]{%
5607   \DTLifint{#1}%
5608   {\edef#2{\char#1\relax}}%
5609   {%
5610     \ifcsundef{#1groupname}{\def#2{#1}}{\letcs#2{#1groupname}}%
5611   }%
5612 }

```

`\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`

```

5613 \newcommand*{\glsgetgrouplabel}[1]{%
5614 \ifthenelse{\equal{#1}{\glssymbolsgroupname}}{\glssymbols}{%
5615 \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`

```

5616 \newcommand*{\setentrycounter}[2][ ]{%
5617   \def\@glo@counterprefix{#1}%
5618   \ifx\@glo@counterprefix\@empty
5619     \def\@glo@counterprefix{.}%
5620   \else

```

```

5621 \def\@glo@counterprefix{.#1.}%
5622 \fi
5623 \def\glstentrycounter{#2}%
5624 }

```

The current glossary style can be set using `\setglossarystyle{<style>}`.

`\setglossarystyle`

```

5625 \newcommand*{\setglossarystyle}[1]{%
5626 \ifcsundef{@glsstyle@#1}%
5627 {%
5628 \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
5629 }%
5630 {%
5631 \csname @glsstyle@#1\endcsname
5632 }%
5633 }

```

`\glossarystyle`

```

5634 \newcommand*{\glossarystyle}[1]{%
5635 \ifcsundef{@glsstyle@#1}%
5636 {%
5637 \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
5638 }%
5639 {%
5640 \GlossariesWarning
5641 {Deprecated command \string\glossarystyle.^^J
5642 I recommend you switch to \string\setglossarystyle\space unless
5643 you want to maintain backward compatibility}%
5644 \setglossentrycompatibility
5645 \csname @glsstyle@#1\endcsname

5646 \ifcsdef{@glscompstyle@#1}%
5647 {\setglossentrycompatibility\csuse{@glscompstyle@#1}}%
5648 {}%
5649 }%
5650 }

```

`\newglossarystyle` New glossary styles can be defined using:

`\newglossarystyle{<name>}{<definition>}`

The *<definition>* argument should redefine the `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.

```

5651 \newcommand{\newglossarystyle}[2]{%
5652 \ifcsundef{@glsstyle@#1}%

```

```

5653  {%
5654    \expandafter\def\csname @glsstyle@#1\endcsname{#2}%
5655  }%
5656  {%
5657    \PackageError{glossaries}{Glossary style ‘#1’ is already defined}{}%
5658  }%
5659 }

```

`\renewglossarystyle` Code for this macro supplied by Marco Daniel.

```

5660 \newcommand{\renewglossarystyle}[2]{%
5661   \ifcsundef{@glsstyle@#1}%
5662   {%
5663     \PackageError{glossaries}{Glossary style ‘#1’ isn’t already defined}{}%
5664   }%
5665   {%
5666     \csdef{@glsstyle@#1}{#2}%
5667   }%
5668 }

```

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`

```

5669 \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`

```

5670 \ifcsundef{hyperlink}%
5671 {%
5672   \def\glshypernumber#1{#1}%

```



```

5673 }%
5674 {%
5675   \def\glshypernumber#1{\@glshypernumber#1\nohyperpage{}}\@nil}
5676 }

```

`\@glshypernumber` This code was provided by Heiko Oberdiek to allow material to be attached to the location.

```

5677 \def\@glshypernumber#1\nohyperpage#2#3\@nil{%
5678   \ifx\#1\%
5679     \else
5680       \@delimR#1\delimR\delimR\%
5681     \fi
5682   \ifx\#2\%
5683     \else
5684       #2%
5685     \fi
5686   \ifx\#3\%
5687     \else
5688       \@glshypernumber#3\@nil
5689     \fi
5690 }

```

`\@delimR` displays a range of numbers for the counter whose name is given by `\@gls@counter` (which must be set prior to using `\glshypernumber`).

`\@delimR`

```

5691 \def\@delimR#1\delimR #2\delimR #3\{%
5692 \ifx\#2\%
5693   \@delimN{#1}%
5694 \else
5695   \@gls@numberlink{#1}\delimR\@gls@numberlink{#2}%
5696 \fi}

```

`\@delimN` displays a list of individual numbers, instead of a range:

`\@delimN`

```

5697 \def\@delimN#1{\@delimN#1\delimN \delimN\%
5698 \def\@delimN#1\delimN #2\delimN#3\{%
5699 \ifx\#3\%
5700   \@gls@numberlink{#1}%
5701 \else
5702   \@gls@numberlink{#1}\delimN\@gls@numberlink{#2}%
5703 \fi
5704 }

```

The following code is modified from hyperref's `\HyInd@pagelink` where the name of the counter being used is given by `\@gls@counter`.

```

5705 \def\@gls@numberlink#1{%
5706 \begingroup

```

```

5707 \toks@={}%
5708 \@gls@removespaces#1 \@nil
5709 \endgroup}

5710 \def\@gls@removespaces#1 #2\@nil{%
5711 \toks@=\expandafter{\the\toks@#1}%
5712 \ifx\#2\%
5713 \edef\x{\the\toks@}%
5714 \ifx\x\empty
5715 \else

5716 \hyperlink{\glstrycounter\@glo@counterprefix\the\toks@}%
5717 {\the\toks@}%
5718 \fi
5719 \else
5720 \@gls@ReturnAfterFi{%
5721 \@gls@removespaces#2\@nil
5722 }%
5723 \fi
5724 }
5725 \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`

```
5726 \newcommand*\hyperrm[1]{\textrm{\glshypernumber{#1}}}
```

`\hypersf`

```
5727 \newcommand*\hypersf[1]{\textsf{\glshypernumber{#1}}}
```

`\hypertt`

```
5728 \newcommand*\hypertt[1]{\texttt{\glshypernumber{#1}}}
```

`\hyperbf`

```
5729 \newcommand*\hyperbf[1]{\textbf{\glshypernumber{#1}}}
```

`\hypermd`

```
5730 \newcommand*\hypermd[1]{\textmd{\glshypernumber{#1}}}
```

`\hyperit`

```
5731 \newcommand*\hyperit[1]{\textit{\glshypernumber{#1}}}
```

`\hypersl`

```
5732 \newcommand*\hypersl[1]{\textsl{\glshypernumber{#1}}}
```

`\hyperup`

```
5733 \newcommand*\hyperup[1]{\textup{\glshypernumber{#1}}}
```

`\hypersc`

```
5734 \newcommand*{\hypersc}[1]{\textsc{\glshypernumber{#1}}}
```

`\hyperemph`

```
5735 \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.

```
5736 \newcommand{\oldacronym}[4][\gls@label]{%
5737   \def\gls@label{#2}%
5738   \newacronym[#4]{#1}{#2}{#3}%
5739   \ifcsundef{xspace}%
5740     {%
5741       \expandafter\edef\csname#1\endcsname{%
5742         \noexpand\@ifstar{\noexpand\Gls{#1}}{\noexpand\gls{#1}}}%
5743     }%
5744   }%
5745   {%
5746     \expandafter\edef\csname#1\endcsname{%
5747       \noexpand\@ifstar{\noexpand\Gls{#1}\noexpand\xspace}{%
5748         \noexpand\gls{#1}\noexpand\xspace}%
5749     }%
5750   }%
5751 }
```

```
\newacronym[⟨key-val list⟩]{⟨label⟩}{⟨abbrev⟩}{⟨long⟩}
```

This is a quick way of defining acronyms, using `\newglossaryentry` with the appropriate values. It sets the glossary type to `\acronymtype` which will be

acronym if the package option `acronym` has been used, otherwise it will be the default glossary. Since `\newacronym` merely calls `\newglossaryentry`, the acronym is treated like any other glossary entry.

If you prefer a different format, you can redefine `\newacronym` as required. The optional argument can be used to override any of the settings.

This is just a stub. It's redefined by commands like `\SetDefaultAcronymStyle`.

`\newacronym`

```
5752 \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.

```
5753 \newcommand*{\acrpluralsuffix}{\glspluralsuffix}
```

If `garamondx` has been loaded, need to use `\textulc` instead of `\textup`.

`\glstextup`

```
5754 \newrobustcmd*{\glstextup}[1]{\ifdef\textulc{\textulc{#1}}{\textup{#1}}}
```

The following are defined for compatibility with version 2.07 and earlier.

`\glsshortkey`

```
5755 \newcommand*{\glsshortkey}{short}
```

`\glsshortpluralkey`

```
5756 \newcommand*{\glsshortpluralkey}{shortplural}
```

`\glslongkey`

```
5757 \newcommand*{\glslongkey}{long}
```

`\glslongpluralkey`

```
5758 \newcommand*{\glslongpluralkey}{longplural}
```

`\acrfull` Full form of the acronym.

```
5759 \newrobustcmd*{\acrfull}{%
```

```
5760 \@ifstar\s@acrfull\ns@acrfull
```

```
5761 }
```

```

5762 \newcommand*\s@acrfull[2][{}]{%
5763   \new@ifnextchar[{\@acrfull{hyper=false,#1}{#2}}{%
5764     {\@acrfull{hyper=false,#1}{#2}[]}%
5765 }
5766 \newcommand*\ns@acrfull[2][{}]{%
5767   \new@ifnextchar[{\@acrfull{#1}{#2}}{%
5768     {\@acrfull{#1}{#2}[]}%
5769 }

```

`\@acrfull` Low-level macro:

```

5770 \def\@acrfull#1#2[#3]{%
    Make it easier for acronym styles to change this:
5771   \acrfullfmt{#1}{#2}{#3}%
5772 }

```

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.

```

5773 \newcommand*\acrfullfmt[3]{%
5774   \acrlinkfullformat{\@acrlong}{\@acrshort}{#1}{#2}{#3}%
5775 }

```

`\acrlinkfullformat` Format for full links like `\acrfull`. Syntax: `\acrlinkfullformat{<long cs>}{<short cs>}{<options>}{<label>}{<insert>}`

```

5776 \newcommand\acrlinkfullformat[5]{%
5777   \acrfullformat{#1}{#3}{#4}[#5]{#2}{#3}{#4}[]}%
5778 }

```

`\acrfullformat` Default full form is `<long>` (`<short>`).

```

5779 \newcommand\acrfullformat[2]{#1\space(#2)}

```

Default format for full acronym

`\Acrfull`

```

5780 \newrobustcmd*\Acrfull{%
5781   \@ifstar\s@Acrfull\ns@Acrfull
5782 }

5783 \newcommand*\s@Acrfull[2][{}]{%
5784   \new@ifnextchar[{\@Acrfull{hyper=false,#1}{#2}}{%
5785     {\@Acrfull{hyper=false,#1}{#2}[]}%
5786 }
5787 \newcommand*\ns@Acrfull[2][{}]{%
5788   \new@ifnextchar[{\@Acrfull{#1}{#2}}{%
5789     {\@Acrfull{#1}{#2}[]}%
5790 }

```

Low-level macro:

```
5791 \def\@Acrfull#1#2[#3]{%
```

Make it easier for acronym styles to change this:

```
5792 \Acrfullfmt{#1}{#2}{#3}%  
5793 }
```

`\Acrfullfmt` First letter upper case full format.

```
5794 \newcommand*\@Acrfullfmt}[3]{%  
5795 \acrlinkfullformat{\@Acrlong}{\@acrshort}{#1}{#2}{#3}%  
5796 }
```

`\ACRfull`

```
5797 \newrobustcmd*\@ACRfull}{%  
5798 \@ifstar\s@ACRfull\@ns@ACRfull  
5799 }  
  
5800 \newcommand*\s@ACRfull[2][]{%  
5801 \new@ifnextchar[{\@ACRfull{hyper=false,#1}{#2}}%  
5802 {\@ACRfull{hyper=false,#1}{#2}[]}%  
5803 }  
5804 \newcommand*\ns@ACRfull[2][]{%  
5805 \new@ifnextchar[{\@ACRfull{#1}{#2}}%  
5806 {\@ACRfull{#1}{#2}[]}%  
5807 }
```

Low-level macro:

```
5808 \def\@ACRfull#1#2[#3]{%
```

Make it easier for acronym styles to change this:

```
5809 \ACRfullfmt{#1}{#2}{#3}%  
5810 }
```

`\ACRfullfmt` All upper case full format.

```
5811 \newcommand*\@ACRfullfmt}[3]{%  
5812 \acrlinkfullformat{\@ACRlong}{\@ACRshort}{#1}{#2}{#3}%  
5813 }
```

Plural:

`\acrfullpl`

```
5814 \newrobustcmd*\@acrfullpl}{%  
5815 \@ifstar\s@acrfullpl\@ns@acrfullpl  
5816 }  
  
5817 \newcommand*\s@acrfullpl[2][]{%  
5818 \new@ifnextchar[{\@acrfullpl{hyper=false,#1}{#2}}%  
5819 {\@acrfullpl{hyper=false,#1}{#2}[]}%  
5820 }  
5821 \newcommand*\ns@acrfullpl[2][]{%
```

```

5822 \new@ifnextchar[{\@acrfullpl{#1}{#2}}%
5823         {\@acrfullpl{#1}{#2} []}%
5824 }

```

Low-level macro:

```

5825 \def\@acrfullpl#1#2[#3]{%

```

Make it easier for acronym styles to change this:

```

5826 \acrfullplfmt{#1}{#2}{#3}%
5827 }

```

`\acrfullplfmt` No case change plural full format.

```

5828 \newcommand*{\acrfullplfmt}[3]{%
5829 \acrlinkfullformat{\@acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%
5830 }

```

`\Acrfullpl`

```

5831 \newrobustcmd*{\Acrfullpl}{%
5832 \@ifstar\s@Acrfullpl\ns@Acrfullpl
5833 }

5834 \newcommand*\s@Acrfullpl[2] []{%
5835 \new@ifnextchar[{\@Acrfullpl{hyper=false,#1}{#2}}%
5836         {\@Acrfullpl{hyper=false,#1}{#2} []}%
5837 }

5838 \newcommand*\ns@Acrfullpl[2] []{%
5839 \new@ifnextchar[{\@Acrfullpl{#1}{#2}}%
5840         {\@Acrfullpl{#1}{#2} []}%
5841 }

```

Low-level macro:

```

5842 \def\@Acrfullpl#1#2[#3]{%

```

Make it easier for acronym styles to change this:

```

5843 \Acrfullplfmt{#1}{#2}{#3}%
5844 }

```

`\Acrfullplfmt` First letter upper case plural full format.

```

5845 \newcommand*{\Acrfullplfmt}[3]{%
5846 \acrlinkfullformat{\@acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%
5847 }

```

`\ACRfullpl`

```

5848 \newrobustcmd*{\ACRfullpl}{%
5849 \@ifstar\s@ACRfullpl\ns@ACRfullpl
5850 }

5851 \newcommand*\s@ACRfullpl[2] []{%
5852 \new@ifnextchar[{\@ACRfullpl{hyper=false,#1}{#2}}%
5853         {\@ACRfullpl{hyper=false,#1}{#2} []}%

```

```

5854 }
5855 \newcommand*\ns@ACRfullpl[2] [] {%
5856   \new@ifnextchar [{\@ACRfullpl{#1}{#2}}}%
5857   {\@ACRfullpl{#1}{#2} []}%
5858 }

```

Low-level macro:

```

5859 \def\@ACRfullpl#1#2[#3] {%
    Make it easier for acronym styles to change this:
5860   \ACRfullplfmt{#1}{#2}{#3}%
5861 }

```

`\ACRfullplfmt` All upper case plural full format.

```

5862 \newcommand*\@ACRfullplfmt[3] {%
5863   \acrlinkfullformat{\@ACRlongpl}{\@ACRshortpl}{#1}{#2}{#3}%
5864 }

```

1.17 Predefined acronym styles

`\acronymfont` This is only used with the additional acronym styles:

```

5865 \newcommand{\acronymfont}[1]{#1}

```

`\firstacronymfont` This is only used with the additional acronym styles:

```

5866 \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.

```

5867 \newcommand*\acrnameformat[2]{\acronymfont{#1}}

```

Define some tokens used by `\newacronym`:

`\glskeylisttok`

```

5868 \newtoks\glskeylisttok

```

`\glslabeltok`

```

5869 \newtoks\glslabeltok

```

`\glsshorttok`

```

5870 \newtoks\glsshorttok

```

`\glslongtok`

```

5871 \newtoks\glslongtok

```

`\newacronymhook` Provide a hook for `\newacronym`:

```

5872 \newcommand*\newacronymhook{}

```


etGenericNewAcronym New improved version of setting the acronym style.

```

5873 \newcommand*{\SetGenericNewAcronym}{%
5874   \renewcommand{\newacronym}[4][{}]{%
5875     \ifdefempty{\@glsacronymlists}%
5876       {%
5877         \def\@glo@type{\acronymtype}%
5878         \setkeys{glossentry}{##1}%
5879         \DeclareAcronymList{\@glo@type}%
5880       }%
5881     }%
5882     \glskeylisttok{##1}%
5883     \glslabeltok{##2}%
5884     \glsshorttok{##3}%
5885     \glslongtok{##4}%
5886     \newacronymhook
5887     \protected@edef\@do@newglossaryentry{%
5888       \noexpand\newglossaryentry{\the\glslabeltok}%
5889       {%
5890         type=\acronymtype,%
5891         name={\expandonce{\acronymentry{##2}}},%
5892         sort={\acronymsort{\the\glsshorttok}{\the\glslongtok}},%
5893         text={\the\glsshorttok},%
5894         short={\the\glsshorttok},%
5895         shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5896         long={\the\glslongtok},%
5897         longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5898         \GenericAcronymFields,%
5899         \the\glskeylisttok
5900       }%
5901     }%
5902     \@do@newglossaryentry
5903   }%

```

Make sure that \acrfull etc reflects the new style:

```

5904 \renewcommand*{\acrfullfmt}[3]{%
5905   \glslink{##1}{##2}{\genacrfullformat{##2}{##3}}}%
5906 \renewcommand*{\Acrfullfmt}[3]{%
5907   \glslink{##1}{##2}{\Genacrfullformat{##2}{##3}}}%
5908 \renewcommand*{\ACRfullfmt}[3]{%
5909   \glslink{##1}{##2}{%
5910     \mfirstucMakeUppercase{\genacrfullformat{##2}{##3}}}%
5911 \renewcommand*{\acrfullplfmt}[3]{%
5912   \glslink{##1}{##2}{\genplacrfullformat{##2}{##3}}}%
5913 \renewcommand*{\Acrfullplfmt}[3]{%
5914   \glslink{##1}{##2}{\Genplacrfullformat{##2}{##3}}}%
5915 \renewcommand*{\ACRfullplfmt}[3]{%
5916   \glslink{##1}{##2}{%
5917     \mfirstucMakeUppercase{\genplacrfullformat{##2}{##3}}}%

```

Make sure that \glsentryfull etc reflects the new style:

```

5918 \renewcommand*{\glsentryfull}[1]{\genacrfullformat{##1}{}}%
5919 \renewcommand*{\Glsentryfull}[1]{\Genacrfullformat{##1}{}}%
5920 \renewcommand*{\glsentryfullpl}[1]{\genplacrfullformat{##1}{}}%
5921 \renewcommand*{\Glsentryfullpl}[1]{\Genplacrfullformat{##1}{}}%
5922 }

```

`\GenericAcronymFields` Fields used by `\SetGenericNewAcronym` that can be changed by the acronym style.

```

5923 \newcommand*{\GenericAcronymFields}{description={\the\glslongtok}}

```

`\acronymentry` `\acronymentry{<label>}`

Display style for the name field in the list of acronyms.

```

5924 \newcommand*{\acronymentry}[1]{\acronymfont{\glsentryshort{#1}}}

```

`\acronymsort` `\acronymsort{<short>}{<long>}`

Default sort format for acronyms.

```

5925 \newcommand*{\acronymsort}[2]{#1}

```

`\setacronymstyle` `\setacronymstyle{<style name>}`

```

5926 \newcommand*{\setacronymstyle}[1]{%
5927   \ifcsundef{@glsacr@dispstyle@#1}%
5928   {%
5929     \PackageError{glossaries}{Undefined acronym style ‘#1’}{%
5930     }%
5931   }%
5932   \ifdefempty{\@glsacronymlists}%
5933   {%
5934     \DeclareAcronymList{\acronymtype}%
5935   }%
5936   {%
5937     \SetGenericNewAcronym
5938     \GlsUseAcrStyleDefs{#1}%
5939     \@for\@gls@type:=\@glsacronymlists\do{%
5940       \defglsentryfmt[\@gls@type]{\GlsUseAcrEntryDispStyle{#1}}%
5941     }%
5942   }%
5943 }

```

`\newacronymstyle` `\newacronymstyle{<style name>}{<entry format definition>}{<display definitions>}`

Defines a new acronym style called *<style name>*.

```

5944 \newcommand*{\newacronymstyle}[3]{%
5945   \ifcsdef{@glsacr@dispstyle@#1}%
5946   {%
5947     \PackageError{glossaries}{Acronym style ‘#1’ already exists}{}%
5948   }%
5949   {%
5950     \csdef{@glsacr@dispstyle@#1}{#2}%
5951     \csdef{@glsacr@styledefs@#1}{#3}%
5952   }%
5953 }
```

\renewacronymstyle Redefines the given acronym style.

```

5954 \newcommand*{\renewacronymstyle}[3]{%
5955   \ifcsdef{@glsacr@dispstyle@#1}%
5956   {%
5957     \csdef{@glsacr@dispstyle@#1}{#2}%
5958     \csdef{@glsacr@styledefs@#1}{#3}%
5959   }%
5960   {%
5961     \PackageError{glossaries}{Acronym style ‘#1’ doesn’t exist}{}%
5962   }%
5963 }
```

seAcrEntryDispStyle

```

5964 \newcommand*{\GlsUseAcrEntryDispStyle}[1]{\csuse{@glsacr@dispstyle@#1}}
```

\GlsUseAcrStyleDefs

```

5965 \newcommand*{\GlsUseAcrStyleDefs}[1]{\csuse{@glsacr@styledefs@#1}}
```

Predefined acronym styles:

long-short *<long>* (*<short>*) acronym style.

```

5966 \newacronymstyle{long-short}%
5967 {%
```

Check for long form in case this is a mixed glossary.

```

5968   \ifglshaslong{\glslabel}{\glsngenacfmt}{\glsngenentryfmt}%
5969 }%
5970 {%
5971   \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
5972   \renewcommand*{\genacrfullformat}[2]{%
5973     \glsentrylong{##1}##2\space
5974     (\protect\firstacronymfont{\glsentryshort{##1}})%
5975   }%
5976   \renewcommand*{\Genacrfullformat}[2]{%
5977     \Glsentrylong{##1}##2\space
5978     (\protect\firstacronymfont{\glsentryshort{##1}})%
5979   }%
```

```

5980 \renewcommand*{\genplacrfullformat}[2]{%
5981   \glentrylongpl{##1}##2\space
5982   (\protect\firstacronymfont{\glentryshortpl{##1}})%
5983 }%
5984 \renewcommand*{\Genplacrfullformat}[2]{%
5985   \Glsentrylongpl{##1}##2\space
5986   (\protect\firstacronymfont{\glentryshortpl{##1}})%
5987 }%
5988 \renewcommand*{\acronymentry}[1]{\acronymfont{\glentryshort{##1}}}
5989 \renewcommand*{\acronymsort}[2]{##1}%
5990 \renewcommand*{\acronymfont}[1]{##1}%
5991 \renewcommand*{\firstacronymfont}[1]{\acronymfont{##1}}%
5992 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
5993 }

```

short-long *<short>* (*<long>*) acronym style.

```

5994 \newacronymstyle{short-long}%
5995 {%
5996   \ifglshaslong{\glslabel}{\glsacronymfont}{\glsentryfont}%
5997 }%
5998 {%
5999   \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
6000   \renewcommand*{\genacrfullformat}[2]{%
6001     \protect\firstacronymfont{\glentryshort{##1}}##2\space
6002     (\glentrylong{##1})%
6003   }%
6004   \renewcommand*{\Genacrfullformat}[2]{%
6005     \protect\firstacronymfont{\Glsentryshort{##1}}##2\space
6006     (\glentrylong{##1})%
6007   }%
6008   \renewcommand*{\genplacrfullformat}[2]{%
6009     \protect\firstacronymfont{\glentryshortpl{##1}}##2\space
6010     (\glentrylongpl{##1})%
6011   }%
6012   \renewcommand*{\Genplacrfullformat}[2]{%
6013     \protect\firstacronymfont{\Glsentryshortpl{##1}}##2\space
6014     (\glentrylongpl{##1})%
6015   }%
6016   \renewcommand*{\acronymentry}[1]{\acronymfont{\glentryshort{##1}}}
6017   \renewcommand*{\acronymsort}[2]{##1}%
6018   \renewcommand*{\acronymfont}[1]{##1}%
6019   \renewcommand*{\firstacronymfont}[1]{\acronymfont{##1}}%
6020   \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
6021 }

```

long-sc-short *<long>* (*\textsc{<short>}*) acronym style.

```

6022 \newacronymstyle{long-sc-short}%

```

```

6023 {%
6024   \GlsUseAcrEntryDispStyle{long-short}%
6025 }%
6026 {%
6027   \GlsUseAcrStyleDefs{long-short}%
6028   \renewcommand{\acronymfont}[1]{\textsc{##1}}%
6029   \renewcommand*{\acrpluralsuffix}{\glstextup{\glspluralsuffix}}%
6030 }

```

long-sm-short *<long>* (\textsmaller{<short>}) acronym style.

```

6031 \newacronymstyle{long-sm-short}%
6032 {%
6033   \GlsUseAcrEntryDispStyle{long-short}%
6034 }%
6035 {%
6036   \GlsUseAcrStyleDefs{long-short}%
6037   \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
6038   \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
6039 }

```

sc-short-long *<short>* (\textsc{<long>}) acronym style.

```

6040 \newacronymstyle{sc-short-long}%
6041 {%
6042   \GlsUseAcrEntryDispStyle{short-long}%
6043 }%
6044 {%
6045   \GlsUseAcrStyleDefs{short-long}%
6046   \renewcommand{\acronymfont}[1]{\textsc{##1}}%
6047   \renewcommand*{\acrpluralsuffix}{\glstextup{\glspluralsuffix}}%
6048 }

```

sm-short-long *<short>* (\textsmaller{<long>}) acronym style.

```

6049 \newacronymstyle{sm-short-long}%
6050 {%
6051   \GlsUseAcrEntryDispStyle{short-long}%
6052 }%
6053 {%
6054   \GlsUseAcrStyleDefs{short-long}%
6055   \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
6056   \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
6057 }

```

long-short-desc *<long>* ({<short>}) acronym style that has an accompanying description (which the user needs to supply).

```

6058 \newacronymstyle{long-short-desc}%
6059 {%
6060   \GlsUseAcrEntryDispStyle{long-short}%
6061 }%
6062 {%

```

```

6063 \GlsUseAcrStyleDefs{long-short}%
6064 \renewcommand*{\GenericAcronymFields}{}%
6065 \renewcommand*{\acronymsort}[2]{##2}%
6066 \renewcommand*{\acronymentry}[1]{%
6067     \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
6068 }

```

long-sc-short-desc *<long>* (\textsc{<short>}) acronym style that has an accompanying description (which the user needs to supply).

```

6069 \newacronymstyle{long-sc-short-desc}%
6070 {%
6071     \GlsUseAcrEntryDisplayStyle{long-sc-short}%
6072 }%
6073 {%
6074     \GlsUseAcrStyleDefs{long-sc-short}%
6075     \renewcommand*{\GenericAcronymFields}{}%
6076     \renewcommand*{\acronymsort}[2]{##2}%
6077     \renewcommand*{\acronymentry}[1]{%
6078         \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
6079 }

```

long-sm-short-desc *<long>* (\textsmaller{<short>}) acronym style that has an accompanying description (which the user needs to supply).

```

6080 \newacronymstyle{long-sm-short-desc}%
6081 {%
6082     \GlsUseAcrEntryDisplayStyle{long-sm-short}%
6083 }%
6084 {%
6085     \GlsUseAcrStyleDefs{long-sm-short}%
6086     \renewcommand*{\GenericAcronymFields}{}%
6087     \renewcommand*{\acronymsort}[2]{##2}%
6088     \renewcommand*{\acronymentry}[1]{%
6089         \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
6090 }

```

short-long-desc *<short>* ({<long>}) acronym style that has an accompanying description (which the user needs to supply).

```

6091 \newacronymstyle{short-long-desc}%
6092 {%
6093     \GlsUseAcrEntryDisplayStyle{short-long}%
6094 }%
6095 {%
6096     \GlsUseAcrStyleDefs{short-long}%
6097     \renewcommand*{\GenericAcronymFields}{}%
6098     \renewcommand*{\acronymsort}[2]{##2}%
6099     \renewcommand*{\acronymentry}[1]{%
6100         \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
6101 }

```

sc-short-long-desc *<long>* (\textsc{<short>}) acronym style that has an accompanying description (which the user needs to supply).

```

6102 \newacronymstyle{sc-short-long-desc}%
6103 {%
6104   \GlsUseAcrEntryDispStyle{sc-short-long}%
6105 }%
6106 {%
6107   \GlsUseAcrStyleDefs{sc-short-long}%
6108   \renewcommand*{\GenericAcronymFields}{}%
6109   \renewcommand*{\acronymsort}[2]{##2}%
6110   \renewcommand*{\acronymentry}[1]{%
6111     \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
6112 }

```

sm-short-long-desc *<long>* (\textsmaller{<short>}) acronym style that has an accompanying description (which the user needs to supply).

```

6113 \newacronymstyle{sm-short-long-desc}%
6114 {%
6115   \GlsUseAcrEntryDispStyle{sm-short-long}%
6116 }%
6117 {%
6118   \GlsUseAcrStyleDefs{sm-short-long}%
6119   \renewcommand*{\GenericAcronymFields}{}%
6120   \renewcommand*{\acronymsort}[2]{##2}%
6121   \renewcommand*{\acronymentry}[1]{%
6122     \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
6123 }

```

dua *<long>* only acronym style.

```

6124 \newacronymstyle{dua}%
6125 {%

```

Check for long form in case this is a mixed glossary.

```

6126   \ifdefempty\glscustomtext
6127   {%
6128     \ifglshaslong{\glslabel}%
6129     {%
6130       \glcifplural
6131       {%

```

Plural form:

```

6132         \glscapscase
6133         {%

```

Plural form, don't adjust case:

```

6134         \glentrylongpl{\glslabel}\glinsert
6135         }%
6136         {%

```

Plural form, make first letter upper case:

```

6137         \Glsentrylongpl{\glslabel}\glsinsert
6138     }%
6139     {%

    Plural form, all caps:
6140         \mfirstucMakeUppercase
6141         {\glsentrylongpl{\glslabel}\glsinsert}%
6142     }%
6143 }%
6144 {%

    Singular form
6145     \glscapscase
6146     {%

    Singular form, don't adjust case:
6147     \glsentrylong{\glslabel}\glsinsert
6148     }%
6149     {%

    Subsequent singular form, make first letter upper case:
6150     \Glsentrylong{\glslabel}\glsinsert
6151     }%
6152     {%

    Subsequent singular form, all caps:
6153     \mfirstucMakeUppercase
6154     {\glsentrylong{\glslabel}\glsinsert}%
6155     }%
6156     }%
6157 }%
6158 {%

    Not an acronym:
6159     \glsgenentryfmt
6160     }%
6161     }%
6162     {\glscustomtext\glsinsert}%
6163 }%
6164 {%
6165     \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%

6166     \renewcommand*{\acrfullfmt}[3]{%
6167         \glslink[##1]{##2}{\glsentrylong{##2}##3\space
6168             (\acronymfont{\glsentryshort{##2}})}}%
6169     \renewcommand*{\Acrfullfmt}[3]{%
6170         \glslink[##1]{##2}{\Glsentrylong{##2}##3\space
6171             (\acronymfont{\glsentryshort{##2}})}}%
6172     \renewcommand*{\ACRfullfmt}[3]{%
6173         \glslink[##1]{##2}{%
6174             \mfirstucMakeUppercase{\glsentrylong{##2}##3\space
6175                 (\acronymfont{\glsentryshort{##2}})}}}%

```



```

6176 \renewcommand*{\acrfullplfmt}[3]{%
6177   \glslink{##1}{##2}{\glsentrylongpl{##2}##3\space
6178     (\acronymfont{\glsentryshortpl{##2}})}}%

6179 \renewcommand*{\Acrfullplfmt}[3]{%
6180   \glslink{##1}{##2}{\Glsentrylongpl{##2}##3\space
6181     (\acronymfont{\glsentryshortpl{##2}})}}%
6182 \renewcommand*{\ACRfullplfmt}[3]{%
6183   \glslink{##1}{##2}{%
6184     \mfirstucMakeUppercase{\glsentrylongpl{##2}##3\space
6185       (\acronymfont{\glsentryshortpl{##2}})}}}%
6186 \renewcommand*{\glsentryfull}[1]{%
6187   \glsentrylong{##1}\space(\acronymfont{\glsentryshort{##1}})%
6188 }%
6189 \renewcommand*{\Glsentryfull}[1]{%
6190   \Glsentrylong{##1}\space(\acronymfont{\glsentryshort{##1}})%
6191 }%
6192 \renewcommand*{\glsentryfullpl}[1]{%
6193   \glsentrylongpl{##1}\space(\acronymfont{\glsentryshortpl{##1}})%
6194 }%
6195 \renewcommand*{\Glsentryfullpl}[1]{%
6196   \Glsentrylongpl{##1}\space(\acronymfont{\glsentryshortpl{##1}})%
6197 }%
6198 \renewcommand*{\acronymentry}[1]{\acronymfont{\glsentryshort{##1}}}%
6199 \renewcommand*{\acronymsort}[2]{##1}%
6200 \renewcommand*{\acronymfont}[1]{##1}%
6201 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
6202 }

```

dua-desc <long> only acronym style with user-supplied description.

```

6203 \newacronymstyle{dua-desc}%
6204 {%
6205   \GlsUseAcrEntryDisplayStyle{dua}%
6206 }%
6207 {%
6208   \GlsUseAcrStyleDefs{dua}%
6209   \renewcommand*{\GenericAcronymFields}{}%

6210   \renewcommand*{\acronymentry}[1]{\acronymfont{\glsentrylong{##1}}}%
6211   \renewcommand*{\acronymsort}[2]{##2}%
6212 }%

```

footnote <short>\footnote{<long>} acronym style.

```

6213 \newacronymstyle{footnote}%
6214 {%

  Check for long form in case this is a mixed glossary.

6215   \ifglshaslong{\glslabel}{\glsngenacfmt}{\glsngenentryfmt}%
6216 }%
6217 {%
6218   \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%

```

Need to ensure hyperlinks are switched off on first use:

```

6219 \glshyperfirstfalse
6220 \renewcommand*{\genacrfullformat}[2]{%
6221 \protect\firstacronymfont{\glentryshort{##1}}##2%
6222 \protect\footnote{\glentrylong{##1}}%
6223 }%
6224 \renewcommand*{\Genacrfullformat}[2]{%
6225 \firstacronymfont{\Glentryshort{##1}}##2%
6226 \protect\footnote{\glentrylong{##1}}%
6227 }%
6228 \renewcommand*{\genplacrfullformat}[2]{%
6229 \protect\firstacronymfont{\glentryshortpl{##1}}##2%
6230 \protect\footnote{\glentrylongpl{##1}}%
6231 }%
6232 \renewcommand*{\Genplacrfullformat}[2]{%
6233 \protect\firstacronymfont{\Glentryshortpl{##1}}##2%
6234 \protect\footnote{\glentrylongpl{##1}}%
6235 }%
6236 \renewcommand*{\acronymentry}[1]{\acronymfont{\glentryshort{##1}}}%
6237 \renewcommand*{\acronymsort}[2]{##1}%
6238 \renewcommand*{\acronymfont}[1]{##1}%
6239 \renewcommand*{\acrpluralsuffix}{\glpluralsuffix}%

```

Don't use footnotes for \acrfull:

```

6240 \renewcommand*{\acrfullfmt}[3]{%
6241 \glslink[##1]{##2}{\acronymfont{\glentryshort{##2}}##3\space
6242 (\glentrylong{##2})}%
6243 \renewcommand*{\Acrfullfmt}[3]{%
6244 \glslink[##1]{##2}{\acronymfont{\Glentryshort{##2}}##3\space
6245 (\glentrylong{##2})}%
6246 \renewcommand*{\ACRfullfmt}[3]{%
6247 \glslink[##1]{##2}{%
6248 \mfirstucMakeUppercase{\acronymfont{\glentryshort{##2}}##3\space
6249 (\glentrylong{##2})}}}%
6250 \renewcommand*{\acrfullplfmt}[3]{%
6251 \glslink[##1]{##2}{\acronymfont{\glentryshortpl{##2}}##3\space
6252 (\glentrylongpl{##2})}%
6253 \renewcommand*{\Acrfullplfmt}[3]{%
6254 \glslink[##1]{##2}{\acronymfont{\Glentryshortpl{##2}}##3\space
6255 (\glentrylongpl{##2})}%
6256 \renewcommand*{\ACRfullplfmt}[3]{%
6257 \glslink[##1]{##2}{%
6258 \mfirstucMakeUppercase{\acronymfont{\glentryshortpl{##2}}##3\space
6259 (\glentrylongpl{##2})}}}%

```

Similarly for \glentryfull etc:

```

6260 \renewcommand*{\glentryfull}[1]{%
6261 \acronymfont{\glentryshort{##1}}\space(\glentrylong{##1})}%
6262 \renewcommand*{\Glentryfull}[1]{%
6263 \acronymfont{\Glentryshort{##1}}\space(\glentrylong{##1})}%

```

```

6264 \renewcommand*{\glsentryfullpl}[1]{%
6265     \acronymfont{\glsentryshortpl{##1}}\space(\glsentrylongpl{##1})}%
6266 \renewcommand*{\Glsentryfullpl}[1]{%
6267     \acronymfont{\Glsentryshortpl{##1}}\space(\glsentrylongpl{##1})}%
6268 }

```

footnote-sc \textsc{<short>}\footnote{<long>} acronym style.

```

6269 \newacronymstyle{footnote-sc}%
6270 {%
6271     \GlsUseAcrEntryDisplayStyle{footnote}%
6272 }%
6273 {%
6274     \GlsUseAcrStyleDefs{footnote}%
6275     \renewcommand{\acronymentry}[1]{\acronymfont{\glsentryshort{##1}}}
6276     \renewcommand{\acronymfont}[1]{\textsc{##1}}%
6277     \renewcommand*{\acrpluralsuffix}{\glstextup{\glspluralsuffix}}%
6278 }%

```

footnote-sm \textsmaller{<short>}\footnote{<long>} acronym style.

```

6279 \newacronymstyle{footnote-sm}%
6280 {%
6281     \GlsUseAcrEntryDisplayStyle{footnote}%
6282 }%
6283 {%
6284     \GlsUseAcrStyleDefs{footnote}%
6285     \renewcommand{\acronymentry}[1]{\acronymfont{\glsentryshort{##1}}}
6286     \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
6287     \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
6288 }%

```

footnote-desc <short>\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```

6289 \newacronymstyle{footnote-desc}%
6290 {%
6291     \GlsUseAcrEntryDisplayStyle{footnote}%
6292 }%
6293 {%
6294     \GlsUseAcrStyleDefs{footnote}%
6295     \renewcommand*{\GenericAcronymFields}{}%
6296     \renewcommand*{\acronymsort}[2]{##2}%
6297     \renewcommand*{\acronymentry}[1]{%
6298         \glsentrylong{##1}\space (\acronymfont{\glsentryshort{##1}})%
6299     }%

```

footnote-sc-desc \textsc{<short>}\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```

6300 \newacronymstyle{footnote-sc-desc}%
6301 {%

```

```

6302 \GlsUseAcrEntryDispStyle{footnote-sc}%
6303 }%
6304 {%
6305 \GlsUseAcrStyleDefs{footnote-sc}%
6306 \renewcommand*{\GenericAcronymFields}{}%
6307 \renewcommand*{\acronymsort}[2]{##2}%
6308 \renewcommand*{\acronymentry}[1]{%
6309 \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
6310 }

```

footnote-sm-desc \textsmaller{<short>}\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```

6311 \newacronymstyle{footnote-sm-desc}%
6312 {%
6313 \GlsUseAcrEntryDispStyle{footnote-sm}%
6314 }%
6315 {%
6316 \GlsUseAcrStyleDefs{footnote-sm}%
6317 \renewcommand*{\GenericAcronymFields}{}%
6318 \renewcommand*{\acronymsort}[2]{##2}%
6319 \renewcommand*{\acronymentry}[1]{%
6320 \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
6321 }

```

fineAcronymSynonyms

```

6322 \newcommand*{\DefineAcronymSynonyms}{%

```

Short form

\acs

```

6323 \let\acs\acrshort

```

First letter uppercase short form

\Acs

```

6324 \let\Acs\Acrshort

```

Plural short form

\acsp

```

6325 \let\acsp\acrshortpl

```

First letter uppercase plural short form

\Acsp

```

6326 \let\Acsp\Acrshortpl

```

Long form

\acl

```

6327 \let\acl\aclong

```

Plural long form

\aclp

6328 \let\aclp\acrlongpl

First letter upper case long form

\Acl

6329 \let\Acl\Acrlong

First letter upper case plural long form

\Ac lp

6330 \let\Ac lp\Acr long pl

Full form

\acf

6331 \let\acf\acrfull

Plural full form

\acfp

6332 \let\acfp\acrfullpl

First letter upper case full form

\Acf

6333 \let\Acf\Acrfull

First letter upper case plural full form

\Acfp

6334 \let\Acfp\Acrfullpl

Standard form

\ac

6335 \let\ac\gls

First upper case standard form

\Ac

6336 \let\Ac\Gls

Standard plural form

\acp

6337 \let\acp\glspl

Standard first letter upper case plural form

\Acp

6338 \let\Acp\Glspl

6339 }

Define synonyms if required

6340 \ifglsacrshortcuts

6341 \DefineAcronymSynonyms

6342 \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.

6343 \newcommand*\SetDefaultAcronymDisplayStyle[1]{%

6344 \defglsentryfmt[#1]{\glsentryfmt}%

6345 }

DefaultNewAcronymDef Sets up the acronym definition for the default style. The information is provided by the tokens \glslabeltok, \glsshorttok, \glslongtok and \glskeylisttok.

6346 \newcommand*\DefaultNewAcronymDef{%

6347 \edef\@do@newglossaryentry{%

6348 \noexpand\newglossaryentry{\the\glslabeltok}%

6349 {%

6350 type=\acronymtype,%

6351 name={\the\glsshorttok},%

6352 sort={\the\glsshorttok},%

6353 text={\the\glsshorttok},%

6354 first={\acrfullformat{\the\glslongtok}{\the\glsshorttok}},%

6355 plural={\noexpand\expandonce\noexpand\@glo@shortpl},%

6356 firstplural={\acrfullformat{\noexpand\expandonce\noexpand\@glo@longpl}%

6357 {\noexpand\expandonce\noexpand\@glo@shortpl}},%

6358 short={\the\glsshorttok},%

6359 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%

6360 long={\the\glslongtok},%

6361 longplural={\the\glslongtok\noexpand\acrpluralsuffix},%

6362 description={\the\glslongtok},%

6363 descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%

Remaining options specified by the user:

6364 \the\glskeylisttok

6365 }%

6366 }%

6367 \let\@org@gls@assign@firstpl\gls@assign@firstpl

6368 \let\@org@gls@assign@plural\gls@assign@plural

6369 \let\@org@gls@assign@descplural\gls@assign@descplural

6370 \def\gls@assign@firstpl##1##2{%

6371 \@@gls@expand@field{##1}{firstpl}{##2}%

6372 }%

6373 \def\gls@assign@plural##1##2{%

6374 \@@gls@expand@field{##1}{plural}{##2}%

6375 }%

```

6376 \def\gls@assign@descplural##1##2{%
6377   \@@gls@expand@field{##1}{descplural}{##2}%
6378 }%
6379 \do@newglossaryentry
6380 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6381 \let\gls@assign@plural\@org@gls@assign@plural
6382 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6383 }

```

DefaultAcronymStyle Set up the default acronym style:

```

6384 \newcommand*\SetDefaultAcronymStyle{%

```

Set the display style:

```

6385 \@for\@gls@type:=\@gls@acronymlists\do{%
6386   \SetDefaultAcronymDisplayStyle{\@gls@type}%
6387 }%

```

Set up the definition of \newacronym:

```

6388 \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).

```

6389   \ifx\@gls@acronymlists\@empty
6390     \def\@glo@type{\acronymtype}%
6391     \setkeys{glossentry}{##1}%
6392     \DeclareAcronymList{\@glo@type}%
6393     \SetDefaultAcronymDisplayStyle{\@glo@type}%
6394   \fi
6395   \glskeylisttok{##1}%
6396   \glslabeltok{##2}%
6397   \glsshorttok{##3}%
6398   \glslongtok{##4}%
6399   \newacronymhook
6400   \DefaultNewAcronymDef
6401 }%
6402 \renewcommand*\acrpluralsuffix{\glspluralsuffix}%
6403 }

```

\acrfootnote Used by the footnote acronym styles.

```

6404 \newcommand*\acrfootnote}[3]{\acrlinkfootnote{#1}{#2}{#3}}

```

\acrlinkfootnote

```

6405 \newcommand*\acrlinkfootnote}[3]{%
6406   \footnote{\glslink{#1}{#2}{#3}}%
6407 }

```

\acrlinkfootnote

```

6408 \newcommand*\acrlinkfootnote}[3]{%
6409   \footnote{#3}%
6410 }

```

AcronymDisplayStyle Sets the acronym display style for given glossary for the description and footnote combination.

```

6411 \newcommand*{\SetDescriptionFootnoteAcronymDisplayStyle}[1]{%
6412   \def\glsentryfmt[#1]{%

6413     \ifdefempty\glscustomtext
6414     {%
6415       \ifglssused{\glslabel}%
6416       {%
6417         \acronymfont{\glsentryfmt}%
6418       }%
6419     {%
6420       \firstacronymfont{\glsentryfmt}%
6421       \ifglsshassymbol{\glslabel}%
6422       {%
6423         \expandafter\protect\expandafter\acrfootnote\expandafter
6424         {\@gls@link@opts}{\@gls@link@label}%
6425       {%
6426         \glsifplural
6427         {\glsentrysymbolplural{\glslabel}}%
6428         {\glsentrysymbol{\glslabel}}%
6429       }%
6430     }%
6431   }%
6432 }%
6433 {\glscustomtext\glsinsert}%
6434 }%
6435 }

```

otnoteNewAcronymDef

```

6436 \newcommand*{\DescriptionFootnoteNewAcronymDef}{%
6437   \edef\@do@newglossaryentry{%
6438     \noexpand\newglossaryentry{\the\glslabeltok}%
6439     {%
6440       type=\acronymtype,%
6441       name={\noexpand\acronymfont{\the\glsshorttok}},%
6442       sort={\the\glsshorttok},%
6443       first={\the\glsshorttok},%
6444       firstplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6445       text={\the\glsshorttok},%
6446       plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6447       short={\the\glsshorttok},%
6448       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6449       long={\the\glslongtok},%
6450       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6451       symbol={\the\glslongtok},%
6452       symbolplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6453       \the\glskeylisttok
6454     }%

```



```

6455 }%
6456 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6457 \let\@org@gls@assign@plural\gls@assign@plural
6458 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6459 \def\gls@assign@firstpl##1##2{%
6460   \@@gls@expand@field{##1}{firstpl}{##2}%
6461 }%
6462 \def\gls@assign@plural##1##2{%
6463   \@@gls@expand@field{##1}{plural}{##2}%
6464 }%
6465 \def\gls@assign@symbolplural##1##2{%
6466   \@@gls@expand@field{##1}{symbolplural}{##2}%
6467 }%
6468 \do@newglossaryentry
6469 \let\gls@assign@plural\@org@gls@assign@plural
6470 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6471 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6472 }

```

footnoteAcronymStyle If a description and footnote are both required, store the long form in the symbol key. Store the short form in text key. Note that since the long form is stored in the symbol key, if you want the long form to appear in the list of acronyms, you need to use a glossary style that displays the symbol key.

```

6473 \newcommand*{\SetDescriptionFootnoteAcronymStyle}{%
6474   \renewcommand{\newacronym}[4][\]{%
6475     \ifx\@glsacronymlists\@empty
6476       \def\@glo@type{\acronymtype}%
6477       \setkeys{glossentry}{##1}%
6478       \DeclareAcronymList{\@glo@type}%
6479       \SetDescriptionFootnoteAcronymDisplayStyle{\@glo@type}%
6480     \fi
6481     \glskeylisttok{##1}%
6482     \glslabeltok{##2}%
6483     \glsshorttok{##3}%
6484     \gslongtok{##4}%
6485     \newacronymhook
6486     \DescriptionFootnoteNewAcronymDef
6487   }%

```

If footnote package option is specified, set the first use to append the long form (stored in symbol) as a footnote.

```

6488 \@for\@gls@type:=\@glsacronymlists\do{%
6489   \SetDescriptionFootnoteAcronymDisplayStyle{\@gls@type}%
6490 }%

```

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.

```

6491 \ifglsacrsmallcaps
6492   \renewcommand*{\acronymfont}[1]{\textsc{##1}}%

```

```

6493 \renewcommand*{\acrpluralsuffix}{%
6494 \glstextup{\glspluralsuffix}}%
6495 \else
6496 \ifglsacrsmaller
6497 \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%
6498 \fi
6499 \fi

```

Check for package option clash

```

6500 \ifglsacrdua
6501 \PackageError{glossaries}{Option clash: ‘footnote’ and ‘dua’
6502 can’t both be set}{}%
6503 \fi
6504 }%

```

AcronymDisplayStyle Sets the acronym display style for given glossary with description and dua combination.

```

6505 \newcommand*{\SetDescriptionDUAAcronymDisplayStyle}[1]{%
6506 \defglsentryfmt[#1]{\glsgenentryfmt}%
6507 }

```

ionDUANewAcronymDef

```

6508 \newcommand*{\DescriptionDUANewAcronymDef}{%
6509 \edef\@do@newglossaryentry{%
6510 \noexpand\newglossaryentry{\the\glslabeltok}%
6511 {%
6512 type=\acronymtype,%
6513 name={\the\glslongtok},%
6514 sort={\the\glslongtok},%
6515 text={\the\glslongtok},%
6516 first={\the\glslongtok},%
6517 plural={\noexpand\expandonce\noexpand\@glo@longpl},%
6518 firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6519 short={\the\glsshorttok},%
6520 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6521 long={\the\glslongtok},%
6522 longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6523 symbol={\the\glsshorttok},%
6524 symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6525 \the\glskeylisttok
6526 }%
6527 }%
6528 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6529 \let\@org@gls@assign@plural\gls@assign@plural
6530 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6531 \def\gls@assign@firstpl##1##2{%
6532 \@@gls@expand@field{##1}{firstpl}{##2}%
6533 }%
6534 \def\gls@assign@plural##1##2{%

```

```

6535 \@@gls@expand@field{##1}{plural}{##2}%
6536 }%
6537 \def\gls@assign@symbolplural##1##2{%
6538 \@@gls@expand@field{##1}{symbolplural}{##2}%
6539 }%
6540 \do@newglossaryentry
6541 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6542 \let\gls@assign@plural\@org@gls@assign@plural
6543 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6544 }

```

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.

```

6545 \newcommand*\SetDescriptionDUAAcronymStyle{%
6546 \ifglsacrsmallcaps
6547 \PackageError{glossaries}{Option clash: 'smallcaps' and 'dua'
6548 can't both be set}{}%
6549 \else
6550 \ifglsacrsmaller
6551 \PackageError{glossaries}{Option clash: 'smaller' and 'dua'
6552 can't both be set}{}%
6553 \fi
6554 \fi
6555 \renewcommand{\newacronym}[4][[]]{%
6556 \ifx\@glsacronymlists\@empty
6557 \def\@glo@type{\acronymtype}%
6558 \setkeys{glossentry}{##1}%
6559 \DeclareAcronymList{\@glo@type}%
6560 \SetDescriptionDUAAcronymDisplayStyle{\@glo@type}%
6561 \fi
6562 \glskeylisttok{##1}%
6563 \glslabeltok{##2}%
6564 \glsshorttok{##3}%
6565 \glslongtok{##4}%
6566 \newacronymhook
6567 \DescriptionDUANewAcronymDef
6568 }%

```

Set display.

```

6569 \@for\@gls@type:=\@glsacronymlists\do{%
6570 \SetDescriptionDUAAcronymDisplayStyle{\@gls@type}%
6571 }%
6572 }%

```

AcronymDisplayStyle Sets the acronym display style for given glossary using the description setting (but not footnote or dua).

```

6573 \newcommand*\SetDescriptionAcronymDisplayStyle[1]{%
6574 \defglsentryfmt[#1]{%

```

```

6575 \ifdefempty\glscustomtext
6576 {%
6577 \ifglssused{\glslabel}%
6578 {%

Move the inserted text outside of \acronymfont

6579 \let\gls@org@insert\glsinsert
6580 \let\glsinsert\@empty
6581 \acronymfont{\glsgenentryfmt}\gls@org@insert
6582 }%
6583 {%
6584 \glsgenentryfmt
6585 \ifglshassymbol{\glslabel}%
6586 {%
6587 \glsifplural
6588 {%
6589 \def\@glo@symbol{\glsentrysymbolplural{\glslabel}}%
6590 }%
6591 {%
6592 \def\@glo@symbol{\glsentrysymbol{\glslabel}}%
6593 }%
6594 \space(\protect\firstacronymfont
6595 {\glscapscase
6596 {\@glo@symbol}
6597 {\@glo@symbol}
6598 {\mfirstucMakeUppercase{\@glo@symbol}}})%
6599 }%
6600 }%
6601 }%
6602 }%
6603 {\glscustomtext\glsinsert}%
6604 }%
6605 }

```

ptionNewAcronymDef

```

6606 \newcommand*{\DescriptionNewAcronymDef}{%
6607 \edef\@do@newglossaryentry{%
6608 \noexpand\newglossaryentry{\the\glslabeltok}%
6609 {%
6610 type=\acronymtype,%
6611 name={\noexpand
6612 \acrnameformat{\the\glsshorttok}{\the\glslongtok}},%
6613 sort={\the\glsshorttok},%
6614 first={\the\glslongtok},%
6615 firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6616 text={\the\glsshorttok},%
6617 plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6618 short={\the\glsshorttok},%
6619 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6620 long={\the\glslongtok},%

```

```

6621     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6622     symbol={\noexpand\@glo@text},%
6623     symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6624     \the\glskeylisttok}%
6625 }%
6626 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6627 \let\@org@gls@assign@plural\gls@assign@plural
6628 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6629 \def\gls@assign@firstpl##1##2{%
6630     \@@gls@expand@field{##1}{firstpl}{##2}%
6631 }%
6632 \def\gls@assign@plural##1##2{%
6633     \@@gls@expand@field{##1}{plural}{##2}%
6634 }%
6635 \def\gls@assign@symbolplural##1##2{%
6636     \@@gls@expand@field{##1}{symbolplural}{##2}%
6637 }%
6638 \do@newglossaryentry
6639 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6640 \let\gls@assign@plural\@org@gls@assign@plural
6641 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6642 }

```

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.

```

6643 \newcommand*{\SetDescriptionAcronymStyle}{%
6644     \renewcommand{\newacronym}[4][[]]{%
6645         \ifx\@glsacronymlists\@empty
6646             \def\@glo@type{\acronymtype}%
6647             \setkeys{glossentry}{##1}%
6648             \DeclareAcronymList{\@glo@type}%
6649             \SetDescriptionAcronymDisplayStyle{\@glo@type}%
6650         \fi
6651         \glskeylisttok{##1}%
6652         \glslabeltok{##2}%
6653         \glsshorttok{##3}%
6654         \glslongtok{##4}%
6655         \newacronymhook
6656         \DescriptionNewAcronymDef
6657     }%

```

Set display.

```

6658     \@for\@gls@type:=\@glsacronymlists\do{%
6659         \SetDescriptionAcronymDisplayStyle{\@gls@type}%
6660     }%

```

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.

```

6661 \ifglsmallcaps
6662 \renewcommand{\acronymfont}[1]{\textsc{##1}}
6663 \renewcommand*{\acrpluralsuffix}{%
6664 \glstextup{\glspluralsuffix}}%
6665 \else
6666 \ifglsmaller
6667 \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%
6668 \fi
6669 \fi
6670 }%

```

AcronymDisplayStyle Sets the acronym display style for given glossary with footnote setting (but not description or dua).

```

6671 \newcommand*{\SetFootnoteAcronymDisplayStyle}[1]{%
6672 \defglentryfmt[#1]{%

```

```

6673 \ifdefempty\glscustomtext
6674 {%

```

Move the inserted text outside of \acronymfont

```

6675 \let\gls@org@insert\glsinsert
6676 \let\glsinsert\@empty
6677 \ifglused{\glslabel}%
6678 {%
6679 \acronymfont{\glsgenentryfmt}\gls@org@insert
6680 }%
6681 {%
6682 \firstacronymfont{\glsgenentryfmt}\gls@org@insert
6683 \ifglshaslong{\glslabel}%
6684 {%
6685 \expandafter\protect\expandafter\acrfootnote\expandafter
6686 {\@gls@link@opts}{\@gls@link@label}%
6687 {%
6688 \glsifplural
6689 {\glstrylongpl{\glslabel}}%
6690 {\glstrylong{\glslabel}}%
6691 }%
6692 }%
6693 }%
6694 }%
6695 }%
6696 {\glscustomtext\glsinsert}%
6697 }%
6698 }

```

FootnoteNewAcronymDef

```

6699 \newcommand*{\FootnoteNewAcronymDef}{%

```

```

6700 \edef\@do@newglossaryentry{%
6701   \noexpand\newglossaryentry{\the\glslabeltok}%
6702   {%
6703     type=\acronymtype,%
6704     name={\noexpand\acronymfont{\the\glsshorttok}},%
6705     sort={\the\glsshorttok},%
6706     text={\the\glsshorttok},%
6707     plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6708     first={\the\glsshorttok},%
6709     firstplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6710     short={\the\glsshorttok},%
6711     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6712     long={\the\glslongtok},%
6713     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6714     description={\the\glslongtok},%
6715     descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6716     \the\glskeylisttok
6717   }%
6718 }%
6719 \let\@org@gls@assign@plural\gls@assign@plural
6720 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6721 \let\@org@gls@assign@descplural\gls@assign@descplural
6722 \def\gls@assign@firstpl##1##2{%
6723   \@@gls@expand@field{##1}{firstpl}{##2}%
6724 }%
6725 \def\gls@assign@plural##1##2{%
6726   \@@gls@expand@field{##1}{plural}{##2}%
6727 }%
6728 \def\gls@assign@descplural##1##2{%
6729   \@@gls@expand@field{##1}{descplural}{##2}%
6730 }%
6731 \@do@newglossaryentry
6732 \let\gls@assign@plural\@org@gls@assign@plural
6733 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6734 \let\gls@assign@descplural\@org@gls@assign@descplural
6735 }

```

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.

```

6736 \newcommand*\SetFootnoteAcronymStyle{%
6737   \renewcommand{\newacronym}[4][{}]{%
6738     \ifx\@glsacronymlists\empty
6739       \def\@glo@type{\acronymtype}%
6740       \setkeys{glossentry}{##1}%
6741       \DeclareAcronymList{\@glo@type}%
6742       \SetFootnoteAcronymDisplayStyle{\@glo@type}%
6743     \fi
6744     \glskeylisttok{##1}%

```

```

6745 \glslabeltok{##2}%
6746 \glsshorttok{##3}%
6747 \gslongtok{##4}%
6748 \newacronymhook
6749 \FootnoteNewAcronymDef
6750 }%

```

Set display

```

6751 \@for\@gls@type:=\@glsacronymlists\do{%
6752 \SetFootnoteAcronymDisplayStyle{\@gls@type}%
6753 }%

```

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.

```

6754 \ifglsacrsmallcaps
6755 \renewcommand*{\acronymfont}[1]{\textsc{##1}}%
6756 \renewcommand*{\acrpluralsuffix}{%
6757 \glstextup{\glspluralsuffix}}%
6758 \else
6759 \ifglsacrsmaller
6760 \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%
6761 \fi
6762 \fi

```

Check for option clash

```

6763 \ifglsacrdua
6764 \PackageError{glossaries}{Option clash: ‘footnote’ and ‘dua’
6765 can’t both be set}{}%
6766 \fi
6767 }%

```

`\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.

```

6768 \DeclareRobustCommand*\glsdoparenifnotempty}[2]{%
6769 \protected@edef\gls@tmp{#1}%
6770 \ifdefempty\gls@tmp
6771 {}%
6772 {%
6773 \ifx\gls@tmp\@gls@default@value
6774 \else
6775 \space (#2{#1})%
6776 \fi
6777 }%
6778 }

```

`AcronymDisplayStyle` Sets the acronym display style for given glossary where neither footnote nor description is required, but smallcaps or smaller specified.

```

6779 \newcommand*\SetSmallAcronymDisplayStyle}[1]{%

```



```

6780 \defglentryfmt[#1]{%
6781 \ifdefempty\glscustomtext
6782 {%
        Move the inserted text outside of \acronymfont
6783 \let\gls@org@insert\glsinsert
6784 \let\glsinsert\@empty
6785 \ifglused{\glslabel}%
6786 {%
6787 \acronymfont{\glsgenentryfmt}\gls@org@insert
6788 }%
6789 {%
6790 \glsgenentryfmt
6791 \ifglshassymbol{\glslabel}%
6792 {%
6793 \glsifplural
6794 {%
6795 \def\@glo@symbol{\glstentrysymbolplural{\glslabel}}%
6796 }%
6797 {%
6798 \def\@glo@symbol{\glstentrysymbol{\glslabel}}%
6799 }%
6800 \space
6801 (\glscapscase
6802 {\firstacronymfont{\@glo@symbol}}%
6803 {\firstacronymfont{\@glo@symbol}}%
6804 {\firstacronymfont{\mfirstucMakeUppercase{\@glo@symbol}}})%
6805 }%
6806 {}%
6807 }%
6808 }%
6809 {\glscustomtext\glsinsert}%
6810 }%
6811 }

```

\SmallNewAcronymDef

```

6812 \newcommand*{\SmallNewAcronymDef}{%
6813 \edef\@do@newglossaryentry{%
6814 \noexpand\newglossaryentry{\the\glslabeltok}%
6815 {%
6816 type=\acronymtype,%
6817 name={\noexpand\acronymfont{\the\glsshorttok}},%
6818 sort={\the\glsshorttok},%
6819 text={\the\glsshorttok},%
        Default to the short plural.
6820 plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6821 first={\the\glslongtok},%

```

Default to the long plural.

```

6822     firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6823     short={\the\glsshorttok},%
6824     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6825     long={\the\glslongtok},%
6826     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6827     description={\noexpand\@glo@first},%
6828     descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6829     symbol={\the\glsshorttok},%

```

Default to the short plural.

```

6830     symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6831     \the\glskeylisttok
6832 }%
6833 }%
6834 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6835 \let\@org@gls@assign@plural\gls@assign@plural
6836 \let\@org@gls@assign@descplural\gls@assign@descplural
6837 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6838 \def\gls@assign@firstpl##1##2{%
6839   \@@gls@expand@field{##1}{firstpl}{##2}%
6840 }%
6841 \def\gls@assign@plural##1##2{%
6842   \@@gls@expand@field{##1}{plural}{##2}%
6843 }%
6844 \def\gls@assign@descplural##1##2{%
6845   \@@gls@expand@field{##1}{descplural}{##2}%
6846 }%
6847 \def\gls@assign@symbolplural##1##2{%
6848   \@@gls@expand@field{##1}{symbolplural}{##2}%
6849 }%
6850 \do@newglossaryentry
6851 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6852 \let\gls@assign@plural\@org@gls@assign@plural
6853 \let\gls@assign@descplural\@org@gls@assign@descplural
6854 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6855 }

```

`\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.

```

6856 \newcommand*\SetSmallAcronymStyle{%
6857   \renewcommand{\newacronym}[4][\]{%
6858     \ifx\@glsacronymlists\empty
6859       \def\@glo@type{\acronymtype}%
6860       \setkeys{glossentry}{##1}%
6861       \DeclareAcronymList{\@glo@type}%
6862       \SetSmallAcronymDisplayStyle{\@glo@type}%
6863     \fi
6864     \glskeylisttok{##1}%

```

```

6865 \glslabeltok{##2}%
6866 \glsshorttok{##3}%
6867 \glslongtok{##4}%
6868 \newacronymhook
6869 \SmallNewAcronymDef
6870 }%

```

Change the display since first only contains long form.

```

6871 \@for\@gls@type:=\@glsacronymlists\do{%
6872 \SetSmallAcronymDisplayStyle{\@gls@type}%
6873 }%

```

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.

```

6874 \ifglsacrsmallcaps
6875 \renewcommand*{\acronymfont}[1]{\textsc{##1}}
6876 \renewcommand*{\acrpluralsuffix}{%
6877 \glstextup{\glspluralsuffix}}%
6878 \else
6879 \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}
6880 \fi

```

check for option clash

```

6881 \ifglsacrdua
6882 \ifglsacrsmallcaps
6883 \PackageError{glossaries}{Option clash: ‘smallcaps’ and ‘dua’
6884 can’t both be set}{}%
6885 \else
6886 \PackageError{glossaries}{Option clash: ‘smaller’ and ‘dua’
6887 can’t both be set}{}%
6888 \fi
6889 \fi
6890 }%

```

\SetDUADisplayStyle Sets the acronym display style for given glossary with dua setting.

```

6891 \newcommand*\SetDUADisplayStyle[1]{%
6892 \defglsentryfmt[#1]{\glsgenentryfmt}%
6893 }

```

\DUANewAcronymDef

```

6894 \newcommand*\DUANewAcronymDef{%
6895 \edef\@do@newglossaryentry{%
6896 \noexpand\newglossaryentry{\the\glslabeltok}%
6897 {%
6898 type=\acronymtype,%
6899 name={\the\glsshorttok},%
6900 text={\the\glslongtok},%
6901 first={\the\glslongtok},%
6902 plural={\noexpand\expandonce\noexpand\@glo@longpl},%

```

```

6903     firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6904     short={\the\glsshorttok},%
6905     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6906     long={\the\glslongtok},%
6907     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6908     description={\the\glslongtok},%
6909     descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6910     symbol={\the\glsshorttok},%
6911     symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6912     \the\glskeylisttok
6913 }%
6914 }%
6915 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6916 \let\@org@gls@assign@plural\gls@assign@plural
6917 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6918 \let\@org@gls@assign@descplural\gls@assign@descplural
6919 \def\gls@assign@firstpl##1##2{%
6920   \@@gls@expand@field{##1}{firstpl}{##2}%
6921 }%
6922 \def\gls@assign@plural##1##2{%
6923   \@@gls@expand@field{##1}{plural}{##2}%
6924 }%
6925 \def\gls@assign@symbolplural##1##2{%
6926   \@@gls@expand@field{##1}{symbolplural}{##2}%
6927 }%
6928 \def\gls@assign@descplural##1##2{%
6929   \@@gls@expand@field{##1}{descplural}{##2}%
6930 }%
6931 \do@newglossaryentry
6932 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6933 \let\gls@assign@plural\@org@gls@assign@plural
6934 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6935 \let\gls@assign@descplural\@org@gls@assign@descplural
6936 }

```

`\SetDUASStyle` Always expand acronyms.

```

6937 \newcommand*\SetDUASStyle{%
6938   \renewcommand{\newacronym}[4][\]{%
6939     \ifx\@glsacronymlists\@empty
6940       \def\@glo@type{\acronymtype}%
6941       \setkeys{glossentry}{##1}%
6942       \DeclareAcronymList{\@glo@type}%
6943       \SetDUADisplayStyle{\@glo@type}%
6944     \fi
6945     \glskeylisttok{##1}%
6946     \glslabeltok{##2}%
6947     \glsshorttok{##3}%
6948     \glslongtok{##4}%
6949     \newacronymhook

```

```

6950 \DUANewAcronymDef
6951 }%
    Set the display
6952 \@for\@gls@type:=\@glsacronymlists\do{%
6953 \SetDUADisplayStyle{\@gls@type}%
6954 }%
6955 }

```

\SetAcronymStyle

```

6956 \newcommand*\SetAcronymStyle{%
6957 \SetDefaultAcronymStyle
6958 \ifglsacrdescription
6959 \ifglsacrfootnote
6960 \SetDescriptionFootnoteAcronymStyle
6961 \else
6962 \ifglsacrdua
6963 \SetDescriptionDUAAcronymStyle
6964 \else
6965 \SetDescriptionAcronymStyle
6966 \fi
6967 \fi
6968 \else
6969 \ifglsacrfootnote
6970 \SetFootnoteAcronymStyle
6971 \else
6972 \ifthenelse{\boolean{glsacrsmalls} \OR
6973 \boolean{glsacrsmalls}}{
6974 {%
6975 \SetSmallAcronymStyle
6976 }%
6977 {%
6978 \ifglsacrdua
6979 \SetDUASyle
6980 \fi
6981 }%
6982 \fi
6983 \fi
6984 }

```

Set the acronym style according to the package options

```
6985 \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.

```

6986 \newcommand*{\SetCustomDisplayStyle}[1]{%
6987   \def\glssentryfmt[#1]{\glsgenentryfmt}%
6988 }

```

CustomAcronymFields

```

6989 \newcommand*{\CustomAcronymFields}{%
6990   name={\the\glssshorttok},%
6991   description={\the\glslongtok},%
6992   first={\noexpand\acrfullformat{\the\glslongtok}{\the\glssshorttok}},%
6993   firstplural={\noexpand\acrfullformat
6994     {\noexpand\glssentrylongpl{\the\glslabeltok}}%
6995     {\noexpand\glssentryshortpl{\the\glslabeltok}}},%

6996   text={\the\glssshorttok},%
6997   plural={\the\glssshorttok\noexpand\acrpluralsuffix}%
6998 }

```

CustomNewAcronymDef

```

6999 \newcommand*{\CustomNewAcronymDef}{%
7000   \protected@edef\@do@newglossaryentry{%
7001     \noexpand\newglossaryentry{\the\glslabeltok}%
7002     {%
7003       type=\acronymtype,%
7004       short={\the\glssshorttok},%
7005       shortplural={\the\glssshorttok\noexpand\acrpluralsuffix},%
7006       long={\the\glslongtok},%
7007       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
7008       user1={\the\glssshorttok},%
7009       user2={\the\glssshorttok\noexpand\acrpluralsuffix},%
7010       user3={\the\glslongtok},%
7011       user4={\the\glslongtok\noexpand\acrpluralsuffix},%
7012       \CustomAcronymFields,%
7013       \the\glskeylisttok
7014     }%
7015   }%
7016   \@do@newglossaryentry
7017 }

```

\SetCustomStyle

```

7018 \newcommand*{\SetCustomStyle}{%
7019   \renewcommand{\newacronym}[4][\]{%
7020     \ifx\@glsacronymlists\@empty
7021       \def\@glo@type{\acronymtype}%
7022       \setkeys{glossentry}{##1}%
7023       \DeclareAcronymList{\@glo@type}%
7024       \SetCustomDisplayStyle{\@glo@type}%
7025     \fi
7026     \glskeylisttok{##1}%
7027     \glslabeltok{##2}%

```

```

7028 \glsshorttok{##3}%
7029 \glslongtok{##4}%
7030 \newacronymhook
7031 \CustomNewAcronymDef
7032 }%

Set the display
7033 \@for\@gls@type:=\@glsacronymlists\do{%
7034 \SetCustomDisplayStyle{\@gls@type}%
7035 }%
7036 }

```

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.

```
7037 \RequirePackage{glossary-hypernav}
```

The styles that use list-like environments. These are not loaded if the `nolist` option is used:

```
7038 \@gls@loadlist
```

The styles that use the `longtable` environment. These are not loaded if the `no-long package` option is used.

```
7039 \@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.

```
7040 \@gls@loadsuper
```

The tree-like styles. These are not loaded if the `notree package` option is used.

```
7041 \@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.

```

7042 \ifx\@glossary@default@style\relax
7043 \else
7044 \setglossarystyle{\@glossary@default@style}
7045 \fi

```

1.19 Debugging Commands

`\showgloparent` `\showgloparent{<label>}`

```

7046 \newcommand*{\showgloparent}[1]{%
7047 \expandafter\show\csname glo@glsdetoklabel{#1}@parent\endcsname

```

7048 }

`\showglolevel` `\showglolevel{<label>}`

```
7049 \newcommand*{\showglolevel}[1]{%
7050   \expandafter\show\csname glo@\glsdetoklabel{#1}@level\endcsname
7051 }
```

`\showglotext` `\showglotext{<label>}`

```
7052 \newcommand*{\showglotext}[1]{%
7053   \expandafter\show\csname glo@\glsdetoklabel{#1}@text\endcsname
7054 }
```

`\showgloplural` `\showgloplural{<label>}`

```
7055 \newcommand*{\showgloplural}[1]{%
7056   \expandafter\show\csname glo@\glsdetoklabel{#1}@plural\endcsname
7057 }
```

`\showglofirst` `\showglofirst{<label>}`

```
7058 \newcommand*{\showglofirst}[1]{%
7059   \expandafter\show\csname glo@\glsdetoklabel{#1}@first\endcsname
7060 }
```

`\showglofirstpl` `\showglofirstpl{<label>}`

```
7061 \newcommand*{\showglofirstpl}[1]{%
7062   \expandafter\show\csname glo@\glsdetoklabel{#1}@firstpl\endcsname
7063 }
```

`\showglotype` `\showglotype{<label>}`

```
7064 \newcommand*{\showglotype}[1]{%
7065   \expandafter\show\csname glo@\glsdetoklabel{#1}@type\endcsname
7066 }
```


\showglocounter \showglocounter{<label>}

```
7067 \newcommand*{\showglocounter}[1]{%
7068   \expandafter\show\csname glo@\glsdetoklabel{#1}@counter\endcsname
7069 }
```

\showglouser \showglouser{<label>}

```
7070 \newcommand*{\showglouser}[1]{%
7071   \expandafter\show\csname glo@\glsdetoklabel{#1}@useri\endcsname
7072 }
```

\showglouserii \showglouserii{<label>}

```
7073 \newcommand*{\showglouserii}[1]{%
7074   \expandafter\show\csname glo@\glsdetoklabel{#1}@userii\endcsname
7075 }
```

\showglouseriii \showglouseriii{<label>}

```
7076 \newcommand*{\showglouseriii}[1]{%
7077   \expandafter\show\csname glo@\glsdetoklabel{#1}@useriii\endcsname
7078 }
```

\showglouseriv \showglouseriv{<label>}

```
7079 \newcommand*{\showglouseriv}[1]{%
7080   \expandafter\show\csname glo@\glsdetoklabel{#1}@useriv\endcsname
7081 }
```

\showglouserv \showglouserv{<label>}

```
7082 \newcommand*{\showglouserv}[1]{%
7083   \expandafter\show\csname glo@\glsdetoklabel{#1}@userv\endcsname
7084 }
```

\showglouservi \showglouservi{<label>}

```
7085 \newcommand*{\showglouservi}[1]{%
7086   \expandafter\show\csname glo@\glsdetoklabel{#1}@uservi\endcsname
7087 }
```

\showgloname \showgloname{<label>}

```
7088 \newcommand*{\showgloname}[1]{%
7089   \expandafter\show\csname glo@\glsdetoklabel{#1}@name\endcsname
7090 }
```

\showglodesc \showglodesc{<label>}

```
7091 \newcommand*{\showglodesc}[1]{%
7092   \expandafter\show\csname glo@\glsdetoklabel{#1}@desc\endcsname
7093 }
```

\showglodescplural \showglodescplural{<label>}

```
7094 \newcommand*{\showglodescplural}[1]{%
7095   \expandafter\show\csname glo@\glsdetoklabel{#1}@descplural\endcsname
7096 }
```

\showglosort \showglosort{<label>}

```
7097 \newcommand*{\showglosort}[1]{%
7098   \expandafter\show\csname glo@\glsdetoklabel{#1}@sort\endcsname
7099 }
```

\showglosymbol \showglosymbol{<label>}

```
7100 \newcommand*{\showglosymbol}[1]{%
7101   \expandafter\show\csname glo@\glsdetoklabel{#1}@symbol\endcsname
7102 }
```

`\showglosymbolplural` `\showglosymbolplural{\langle label \rangle}`

```
7103 \newcommand*{\showglosymbolplural}[1]{%  
7104   \expandafter\show\csname glo@\glsdetoklabel{#1}@symbolplural\endcsname  
7105 }
```

`\showgloshort` `\showgloshort{\langle label \rangle}`

```
7106 \newcommand*{\showgloshort}[1]{%  
7107   \expandafter\show\csname glo@\glsdetoklabel{#1}@short\endcsname  
7108 }
```

`\showglolong` `\showglolong{\langle label \rangle}`

```
7109 \newcommand*{\showglolong}[1]{%  
7110   \expandafter\show\csname glo@\glsdetoklabel{#1}@long\endcsname  
7111 }
```

`\showgloindex` `\showgloindex{\langle label \rangle}`

```
7112 \newcommand*{\showgloindex}[1]{%  
7113   \expandafter\show\csname glo@\glsdetoklabel{#1}@index\endcsname  
7114 }
```

`\showgloflag` `\showgloflag{\langle label \rangle}`

```
7115 \newcommand*{\showgloflag}[1]{%  
7116   \expandafter\show\csname ifglo@\glsdetoklabel{#1}@flag\endcsname  
7117 }
```

`\showgloloclist` `\showgloloclist{\langle label \rangle}`

```
7118 \newcommand*{\showgloloclist}[1]{%  
7119   \expandafter\show\csname glo@\glsdetoklabel{#1}@loclist\endcsname  
7120 }
```

`\showacronymlists`

`\showacronymlists`

Show list of glossaries that have been flagged as a list of acronyms.

```
7121 \newcommand*{\showacronymlists}{%  
7122   \show\@glsacronymlists  
7123 }
```

`\showglossaries`

`\showglossaries`

Show list of defined glossaries.

```
7124 \newcommand*{\showglossaries}{%  
7125   \show\@glo@types  
7126 }
```

`\showglossaryin`

`\showglossaryin{<glossary-label>}`

Show the ‘in’ extension for the given glossary.

```
7127 \newcommand*{\showglossaryin}[1]{%  
7128   \expandafter\show\csname @glo@type@#1@in\endcsname  
7129 }
```

`\showglossaryout`

`\showglossaryout{<glossary-label>}`

Show the ‘out’ extension for the given glossary.

```
7130 \newcommand*{\showglossaryout}[1]{%  
7131   \expandafter\show\csname @glo@type@#1@out\endcsname  
7132 }
```

`\showglossarytitle`

`\showglossarytitle{<glossary-label>}`

Show the title for the given glossary.

```
7133 \newcommand*{\showglossarytitle}[1]{%  
7134   \expandafter\show\csname @glo@type@#1@title\endcsname  
7135 }
```

`\showglossarycounter`

`\showglossarycounter{<glossary-label>}`

Show the counter for the given glossary.

```
7136 \newcommand*{\showglossarycounter}[1]{%  
7137   \expandafter\show\csname @glo@type@#1@counter\endcsname  
7138 }
```

```
\showglossaryentries \showglossaryentries{\glossary-label}
```

Show the list of entry labels for the given glossary.

```
7139 \newcommand*{\showglossaryentries}[1]{%
7140   \expandafter\show\csname glolist@#1\endcsname
7141 }
```

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.

```
7142 \csname ifglscpatible-2.07\endcsname
7143   \RequirePackage{glossaries-compatible-207}
7144 \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.

```
7145 \NeedsTeXFormat{LaTeX2e}
7146 \ProvidesPackage{glossaries-prefix}[2013/11/14 v4.0 (NLCT)]
```

Pass all options to glossaries:

```
7147 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{glossaries}}
```

Process options:

```
7148 \ProcessOptions
```

Load glossaries:

```
7149 \RequirePackage{glossaries}
```

Add the new keys:

```
7150 \define@key{glossentry}{prefixfirst}{\def\@glo@entryprefixfirst{#1}}%
7151 \define@key{glossentry}{prefixfirstplural}{\def\@glo@entryprefixfirstplural{#1}}%
7152 \define@key{glossentry}{prefix}{\def\@glo@entryprefix{#1}}%
7153 \define@key{glossentry}{prefixplural}{\def\@glo@entryprefixplural{#1}}%
```

Add them to \@gls@keymap:

```

7154 \appto\@gls@keymap{,%
7155   {prefixfirst}{prefixfirst},%
7156   {prefixfirstplural}{prefixfirstplural},%
7157   {prefix}{prefix},%
7158   {prefixplural}{prefixplural}}%
7159 }

```

Set the default values:

```

7160 \appto\@newglossaryentryprehook{%
7161   \def\@glo@entryprefix{}%
7162   \def\@glo@entryprefixplural{}%
7163   \let\@glo@entryprefixfirst\@gls@default@value
7164   \let\@glo@entryprefixfirstplural\@gls@default@value
7165 }

```

Set the assignment code:

```

7166 \appto\@newglossaryentryposthook{%
7167   \gls@assign@field{}\@glo@label{prefix}\@glo@entryprefix}%
7168   \gls@assign@field{}\@glo@label{prefixplural}\@glo@entryprefixplural}%

```

If prefixfirst has not been supplied, make it the same as prefix.

```

7169   \expandafter\gls@assign@field\expandafter
7170     {\csname glo@\@glo@label @prefix\endcsname}\@glo@label{prefixfirst}%
7171     {\@glo@entryprefixfirst}%

```

If prefixfirstplural has not been supplied, make it the same as prefixplural.

```

7172   \expandafter\gls@assign@field\expandafter
7173     {\csname glo@\@glo@label @prefixplural\endcsname}\@glo@label}%
7174     {prefixfirstplural}\@glo@entryprefixfirstplural}%
7175 }

```

Define commands to access these fields:

glsentryprefixfirst

```

7176 \newcommand*\glsentryprefixfirst[1]{\csuse{glo@#1@prefixfirst}}

```

glsentryprefixfirstplural

```

7177 \newcommand*\glsentryprefixfirstplural[1]{\csuse{glo@#1@prefixfirstplural}}

```

\glsentryprefix

```

7178 \newcommand*\glsentryprefix[1]{\csuse{glo@#1@prefix}}

```

\glsentryprefixplural

```

7179 \newcommand*\glsentryprefixplural[1]{\csuse{glo@#1@prefixplural}}

```

Now for the initial upper case variants:

Glsentryprefixfirst

```

7180 \newrobustcmd*\Glsentryprefixfirst[1]{%
7181   \protected@edef\@glo@text{\csname glo@#1@prefixfirst\endcsname}%
7182   \xmakefirstuc\@glo@text
7183 }

```

ryprefixfirstplural

```
7184 \newrobustcmd*{\Glsentryprefixfirstplural}[1]{%
7185   \protected@edef\@glo@text{\csname glo@#1@prefixfirstplural\endcsname}%
7186   \xmakefirstuc\@glo@text
7187 }
```

\Glsentryprefix

```
7188 \newrobustcmd*{\Glsentryprefix}[1]{%
7189   \protected@edef\@glo@text{\csname glo@#1@prefix\endcsname}%
7190   \xmakefirstuc\@glo@text
7191 }
```

lsentryprefixplural

```
7192 \newrobustcmd*{\Glsentryprefixplural}[1]{%
7193   \protected@edef\@glo@text{\csname glo@#1@prefixplural\endcsname}%
7194   \xmakefirstuc\@glo@text
7195 }
```

Define commands to determine if the prefix keys have been set:

\ifglshasprefix

```
7196 \newcommand*{\ifglshasprefix}[3]{%
7197   \ifcempty{glo@#1@prefix}%
7198   {#3}%
7199   {#2}%
7200 }
```

ifglshasprefixplural

```
7201 \newcommand*{\ifglshasprefixplural}[3]{%
7202   \ifcempty{glo@#1@prefixplural}%
7203   {#3}%
7204   {#2}%
7205 }
```

ifglshasprefixfirst

```
7206 \newcommand*{\ifglshasprefixfirst}[3]{%
7207   \ifcempty{glo@#1@prefixfirst}%
7208   {#3}%
7209   {#2}%
7210 }
```

asprefixfirstplural

```
7211 \newcommand*{\ifglshasprefixfirstplural}[3]{%
7212   \ifcempty{glo@#1@prefixfirstplural}%
7213   {#3}%
7214   {#2}%
7215 }
```

Define commands that insert the prefix before commands like \gls:

\pgls

```
7216 \newrobustcmd{\pgls}{\@ifstar\@spgls\@pgls}
```

\@spgls Starred version.

```
7217 \newcommand*{\@spgls}[2][\@pgls@{hyper=false,#1}{#2}]
```

\@pgls Unstarred version.

```
7218 \newcommand*{\@pgls}[2][\%
7219 \new@ifnextchar[%
7220 {\@pgls@{#1}{#2}}%
7221 {\@pgls@{#1}{#2}[]}%
7222 }
```

\@pgls@ Read in the final optional argument:

```
7223 \def\@pgls@#1#2[#3]{%
7224 \glsdoifexists{#2}%
7225 {%
7226 \ifglsused{#2}%
7227 {%
7228 \glsentryprefix{#2}%
7229 }%
7230 {%
7231 \glsentryprefixfirst{#2}%
7232 }%
7233 \@gls@{#1}{#2}[#3]%
7234 }%
7235 }
```

Similarly for the plural version:

\pglsp1

```
7236 \newrobustcmd{\pglsp1}{\@ifstar\@spglsp1\@pglsp1}
```

\@spglsp1 Starred version.

```
7237 \newcommand*{\@spglsp1}[2][\@pglsp1@{hyper=false,#1}{#2}]
```

\@pglsp1 Unstarred version.

```
7238 \newcommand*{\@pglsp1}[2][\%
7239 \new@ifnextchar[%
7240 {\@pglsp1@{#1}{#2}}%
7241 {\@pglsp1@{#1}{#2}[]}%
7242 }
```

\@pglsp1@ Read in the final optional argument:

```
7243 \def\@pglsp1@#1#2[#3]{%
7244 \glsdoifexists{#2}%
7245 {%
7246 \ifglsused{#2}%
```



```

7247   {%
7248   \glstryprefixplural{#2}%
7249   }%
7250   {%
7251   \glstryprefixfirstplural{#2}%
7252   }%
7253   \@glsp1@{#1}{#2}[#3]%
7254   }%
7255 }

```

Now for the first letter upper case versions:

\Pgls

```
7256 \newrobustcmd{\Pgls}{\@ifstar\@sPgls\@Pgls}
```

\@sPgls Starred version.

```
7257 \newcommand*{\@sPgls}[2][\@Pgls@{hyper=false,#1}{#2}]
```

\@Pgls Unstarred version.

```

7258 \newcommand*{\@Pgls}[2][\%
7259   \new@ifnextchar[\%
7260   {\@Pgls@{#1}{#2}}\%
7261   {\@Pgls@{#1}{#2}}\%
7262 ]

```

\@Pgls@ Read in the final optional argument:

```

7263 \def\@Pgls@#1#2[#3]{\%
7264   \glstryexists{#2}%
7265   {%
7266     \ifglstryused{#2}%
7267     {%
7268       \ifglstryhasprefix{#2}%
7269       {%
7270         \glstryprefix{#2}%
7271         \@glstry@{#1}{#2}[#3]%
7272       }%
7273       {\@glstry@{#1}{#2}[#3]}%
7274     }%
7275     {%
7276       \ifglstryhasprefixfirst{#2}%
7277       {%
7278         \glstryprefixfirst{#2}%
7279         \@glstry@{#1}{#2}[#3]%
7280       }%
7281       {\@glstry@{#1}{#2}[#3]}%
7282     }%
7283   }%
7284 }

```

Similarly for the plural version:

```

\Plspl
7285 \newrobustcmd{\Plspl}{\@ifstar\@sPlspl\@Plspl}

\@sPlspl  Starred version.
7286 \newcommand*{\@sPlspl}[2][\@Plspl@{hyper=false,#1}{#2}]

\@Plspl  Unstarred version.
7287 \newcommand*{\@Plspl}[2][\%
7288   \new@ifnextchar[\%
7289   {\@Plspl@{#1}{#2}}\%
7290   {\@Plspl@{#1}{#2}}\%
7291 ]

\@Plspl@  Read in the final optional argument:
7292 \def\@Plspl@#1#2[#3]{\%
7293   \glsdoifexists{#2}\%
7294   {\%
7295     \ifglshasprefixplural{#2}\%
7296     {\%
7297       \ifglshasprefixplural{#2}\%
7298       {\%
7299         \Glsentryprefixplural{#2}\%
7300         \@Plspl@{#1}{#2}[#3]\%
7301       }\%
7302       {\@Plspl@{#1}{#2}[#3]}\%
7303     }\%
7304     {\%
7305       \ifglshasprefixfirstplural{#2}\%
7306       {\%
7307         \Glsentryprefixfirstplural{#2}\%
7308         \@Plspl@{#1}{#2}[#3]\%
7309       }\%
7310       {\@Plspl@{#1}{#2}[#3]}\%
7311     }\%
7312   }\%
7313 }

```

Finally the all upper case versions:

```

\PGLS
7314 \newrobustcmd{\PGLS}{\@ifstar\@sPGLS\@PGLS}

\@sPGLS  Starred version.
7315 \newcommand*{\@sPGLS}[2][\@PGLS@{hyper=false,#1}{#2}]

```

\@PGLS Unstarred version.

```
7316 \newcommand*{\@PGLS}[2][\%  
7317   \new@ifnextchar[\  
7318   {\@PGLS@{#1}{#2}}%  
7319   {\@PGLS@{#1}{#2}[]}%  
7320 }
```

\@PGLS@ Read in the final optional argument:

```
7321 \def\@PGLS@#1#2[#3]{%  
7322   \glsdoifexists{#2}%  
7323   {%  
7324     \ifglsused{#2}%  
7325     {%  
7326       \mfirstucMakeUppercase{\glsentryprefix{#2}}%  
7327     }%  
7328     {%  
7329       \mfirstucMakeUppercase{\glsentryprefixfirst{#2}}%  
7330     }%  
7331     \@GLS@{#1}{#2}[#3]%  
7332   }%  
7333 }
```

Plural version:

\PGLSp1

```
7334 \newrobustcmd{\PGLSp1}{\@ifstar\@sPGLSp1\@PGLSp1}
```

\@sPGLSp1 Starred version.

```
7335 \newcommand*{\@sPGLSp1}[2][\{\@PGLSp1@{hyper=false,#1}{#2}\}
```

\@PGLSp1 Unstarred version.

```
7336 \newcommand*{\@PGLSp1}[2][\%  
7337   \new@ifnextchar[\  
7338   {\@PGLSp1@{#1}{#2}}%  
7339   {\@PGLSp1@{#1}{#2}[]}%  
7340 }
```

\@PGLSp1@ Read in the final optional argument:

```
7341 \def\@PGLSp1@#1#2[#3]{%  
7342   \glsdoifexists{#2}%  
7343   {%  
7344     \ifglsused{#2}%  
7345     {%  
7346       \mfirstucMakeUppercase{\glsentryprefixplural{#2}}%  
7347     }%  
7348     {%  
7349       \mfirstucMakeUppercase{\glsentryprefixfirstplural{#2}}%  
7350     }%  
7351     \@GLSp1@{#1}{#2}[#3]%  
7352   }%  
7353 }
```

```

7352 }%
7353 }

```

3 Mfirstuc Documented Code

```

7354 \NeedsTeXFormat{LaTeX2e}
7355 \ProvidesPackage{mfirstuc}[2013/11/04 v1.08 (NLCT)]

```

Requires etoolbox:

```

7356 \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.

```

7357 \newif\if@glscs
7358 \newtoks\@glsmfirst
7359 \newtoks\@glsmrest
7360 \newrobustcmd*{\makefirstuc}[1]{%
7361   \def\gls@argi{#1}%
7362   \ifx\gls@argi\@empty

      If the argument is empty, do nothing.

7363   \else

7364     \def\@gls@tmp{\ #1}%
7365     \@onelevel@sanitize\@gls@tmp
7366     \expandafter\@gls@checkcs\@gls@tmp\relax\relax
7367     \if@glscs
7368       \@gls@getbody #1{}\@nil
7369       \ifx\@gls@rest\@empty
7370         \glsmakefirstuc{#1}%
7371       \else
7372         \expandafter\@gls@split\@gls@rest\@nil
7373         \ifx\@gls@first\@empty
7374           \glsmakefirstuc{#1}%
7375         \else
7376           \expandafter\@glsmfirst\expandafter{\@gls@first}%
7377           \expandafter\@glsmrest\expandafter{\@gls@rest}%
7378           \edef\@gls@domfirstuc{\noexpand\@gls@body
7379             {\noexpand\glsmakefirstuc\the\@glsmfirst}%
7380             \the\@glsmrest}%
7381           \@gls@domfirstuc
7382         \fi

```

```

7383     \fi
7384     \else
7385         \glsmakefirstuc{#1}%
7386     \fi
7387 \fi
7388 }

```

Put first argument in \@gls@first and second argument in \@gls@rest:

```

7389 \def\@gls@split#1#2\@nil{%
7390     \def\@gls@first{#1}\def\@gls@rest{#2}%
7391 }

7392 \def\@gls@checkcs#1 #2#3\relax{%
7393     \def\@gls@argi{#1}\def\@gls@argii{#2}%
7394     \ifx\@gls@argi\@gls@argii
7395         \@glscstrue
7396     \else
7397         \@glscsfalse
7398     \fi
7399 }

```

\@gls@makefirstuc Make first thing upper case:

```

7400 \def\@gls@makefirstuc#1{\mfirstucMakeUppercase #1}

```

mfirstucMakeUppercase Allow user to replace \MakeUppercase with another case changing command.

```

7401 \newcommand*{\mfirstucMakeUppercase}{\MakeUppercase}

```

\glsmakefirstuc Provide a user command to make it easier to customise.

```

7402 \newcommand*{\glsmakefirstuc}[1]{\@gls@makefirstuc{#1}}

```

Get the first grouped argument and stores in \@gls@body.

```

7403 \def\@gls@getbody#1#{\def\@gls@body{#1}\@gls@gobbletonil}

```

Scoop up everything to \@nil and store in \@gls@rest:

```

7404 \def\@gls@gobbletonil#1\@nil{\def\@gls@rest{#1}}

```

\xmakefirstuc Expand argument once before applying \makefirstuc (added v1.01).

```

7405 \newcommand*{\xmakefirstuc}[1]{%
7406     \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).

```

7407 \newrobustcmd*{\capitalisewords}[1]{%
7408     \def\gls@add@space{}%
7409     \mfu@capitalisewords#1 \@nil\mfu@endcap
7410 }

```

```

7411 \def\mfu@capitalisewords#1 #2\mfu@endcap{%
7412   \def\mfu@cap@first{#1}%
7413   \def\mfu@cap@second{#2}%
7414   \gls@add@space
7415   \makefirstuc{#1}%
7416   \def\gls@add@space{ }%
7417   \ifx\mfu@cap@second\@nnil
7418     \let\next\mfu@cap\mfu@noop
7419   \else
7420     \let\next\mfu@cap\mfu@capitalisewords
7421   \fi
7422   \next\mfu@cap#2\mfu@endcap
7423 }
7424 \def\mfu@noop#1\mfu@endcap{}

```

`\xcapitalisewords` Short-cut command:

```

7425 \newcommand*{\xcapitalisewords}[1]{%
7426   \expandafter\capitalisewords\expandafter{#1}%
7427 }

```

4 Glossary Styles

4.1 Glossary hyper-navigation definitions (glossary-hypernav package)

Package Definition:

```

7428 \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 hypertext 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`

```

7429 \newcommand*{\glsnavhyperlink}[3][\@glo@type]{%
7430   \edef\gls@grplabel{#2}\protected@edef\@gls@grptitle{#3}%
7431   \@glslink{glsn:#1@#2}{#3}}

```

`\glsnavhypertarget[⟨type⟩]{⟨label⟩}{⟨text⟩}`

This command makes `⟨text⟩` a hypertext 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.

`\glshnavhypertarget`

```
7432 \newcommand*{\glshnavhypertarget}[3][\@gls@type]{%
```

Add this group to the aux file for re-run check.

```
7433 \protected@write\@auxout{}\string\@gls@hypergroup{#1}{#2}}%
```

Add the target.

```
7434 \@glstarget{glsn:#1@#2}{#3}%
```

Check list of know groups to determine if a re-run is required.

```
7435 \expandafter\let
```

```
7436 \expandafter\@gls@list\csname @gls@hypergroup@list@#1\endcsname
```

Iterate through list and terminate loop if this group is found.

```
7437 \@for\@gls@elem:=\@gls@list\do{%
```

```
7438 \ifthenelse{\equal{\@gls@elem}{#2}}{\@endfortrue}{}}%
```

Check if list terminated prematurely.

```
7439 \if@endfor
```

```
7440 \else
```

This group was not included in the list, so issue a warning.

```
7441 \GlossariesWarningNoLine{Navigation panel
```

```
7442 for glossary type ‘#1’^^Jmissing group ‘#2’}%
```

```
7443 \gdef\@gls@hypergroup@rerun{%
```

```
7444 \GlossariesWarningNoLine{Navigation panel
```

```
7445 has changed. Rerun LaTeX}}%
```

```
7446 \fi
```

```
7447 }
```

`\@gls@hypergroup@rerun` Give a warning at the end if re-run required

```
7448 \let\@gls@hypergroup@rerun\relax
```

```
7449 \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.

```
7450 \newcommand*{\@gls@hypergroup}[2]{%
```

```
7451 \@ifundefined{\@gls@hypergroup@list@#1}{%
```

```
7452 \expandafter\xdef\csname @gls@hypergroup@list@#1\endcsname{#2}}%
```

```
7453 }{%
```

```
7454 \expandafter\let\expandafter\@gls@tmp
```

```
7455 \csname @gls@hypergroup@list@#1\endcsname
```

```
7456 \expandafter\xdef\csname @gls@hypergroup@list@#1\endcsname{%
```

```
7457 \@gls@tmp,#2}}%
```

```
7458 }%
```

```
7459 }
```

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`

```

7460 \newcommand*{\glsnavigation}{%
7461 \def\@gls@between{%
7462 \@ifundefined{\@gls@hypergroup@list@{\@glo@type}}{%
7463   \def\@gls@list{%
7464 }{%
7465   \expandafter\let\expandafter\@gls@list
7466     \csname \@gls@hypergroup@list@{\@glo@type}\endcsname
7467 }%
7468 \@for\@gls@tmp:=\@gls@list\do{%
7469   \@gls@between

7470   \@gls@getgrouptitle{\@gls@tmp}{\@gls@grptitle}%
7471   \glsnavhyperlink{\@gls@tmp}{\@gls@grptitle}%
7472   \let\@gls@between\glshypernavsep%
7473 }%
7474 }
```

`\glshypernavsep` Separator for the hyper navigation bar.

```

7475 \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`

```

7476 \newcommand*{\glssymbolnav}{%
7477 \glsnavhyperlink{glssymbols}{\glsgetgrouptitle{glssymbols}}%
7478 \glshypernavsep
7479 \glsnavhyperlink{glsnumbers}{\glsgetgrouptitle{glsnumbers}}%
7480 \glshypernavsep
7481 }
```

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.

```

7482 \ProvidesPackage{glossary-inline}[2013/11/14 v4.0 (NLCT)]
```

`inline` Define the inline style.

```

7483 \newglossarystyle{inline}{%
```


Start of glossary sets up first empty separator between entries. (This is then changed by `\glossentry`)

```
7484 \renewenvironment{theglossary}%
7485   {%
7486     \def\gls@inlinesep{}%
7487     \def\gls@inlinesubsep{}%
7488     \def\gls@inlinepostchild{}%
7489   }%
7490   {\glspostinline}%
```

No header:

```
7491 \renewcommand*{\glossaryheader}{}%
```

No group headings (if heading is required, add `\glsinlinedopostchild` to start definition in case heading follows a child entry):

```
7492 \renewcommand*{\glsgroupheading}[1]{}%
```

Just display separator followed by name and description:

```
7493 \renewcommand{\glossentry}[2] {%
7494   \glsinlinedopostchild
7495   \gls@inlinesep
7496   \glsentryitem{##1}%
7497   \glsinlinenameformat{##1}{%
7498     \glossentryname{##1}%
7499   }%
7500   \ifglshasdescsuppressed{##1}%
7501   {%
7502     \glsinlineemptydescformat
7503     {%
7504       \glossentrysymbol{##1}%
7505     }%
7506     {%
7507       ##2%
7508     }%
7509   }%
7510   {%
7511     \ifglshasdesc{##1}%
7512     {\glsinlinedescformat{\glossentrydesc{##1}}{\glossentrysymbol{##1}}{##2}}%
7513     {\glsinlineemptydescformat{\glossentrysymbol{##1}}{##2}}%
7514   }%
7515   \ifglshaschildren{##1}%
7516   {%
7517     \glsresetsubentrycounter
7518     \glsinlineparentchildseparator
7519     \def\gls@inlinesubsep{}%
7520     \def\gls@inlinepostchild{\glsinlinepostchild}%
7521   }%
7522   {}%
7523   \def\gls@inlinesep{\glsinlineseparator}%
7524 }%
```

Sub-entries display description:

```
7525 \renewcommand{\subglossentry}[3]{%
7526   \gls@inlinesubsep%
7527   \glsinlinesubnameformat{##2}{%
7528     \glossentryname{##2}}%
7529   \glssubentryitem{##2}%
7530   \glsinlinesubdescformat{\glossentrydesc{##2}}{\glossentrysymbol{##2}}{##3}%
7531   \def\gls@inlinesubsep{\glsinlinesubseparator}%
7532 }%
```

Nothing special between groups:

```
7533 \renewcommand*{\glsgroupskip}{}%
7534 }
```

`\glsinlinedopostchild`

```
7535 \newcommand*{\glsinlinedopostchild}{%
7536   \gls@inlinepostchild
7537   \def\gls@inlinepostchild{}%
7538 }
```

`\glsinlineseparator` Separator to use between entries.

```
7539 \newcommand*{\glsinlineseparator}{;\space}
```

`\glsinlinesubseparator` Separator to use between sub-entries.

```
7540 \newcommand*{\glsinlinesubseparator}{,\space}
```

`\glsinlinparentchildseparator` Separator to use between parent and children.

```
7541 \newcommand*{\glsinlinparentchildseparator}{:\space}
```

`\glsinlinepostchild` Hook to use between child and next entry

```
7542 \newcommand*{\glsinlinepostchild}{}
```

`\glsinlinepost` Terminator for inline glossary.

```
7543 \newcommand*{\glsinlinepost}{\glsinlinepostdescription\space}
```

`\glsinlinenameformat` Formats the name of the entry (first argument label, second argument name):

```
7544 \newcommand*{\glsinlinenameformat}[2]{\glstarget{#1}{#2}}
```

`\glsinlinedescformat` Formats the entry's description, symbol and location list:

```
7545 \newcommand*{\glsinlinedescformat}[3]{\space#1}
```

`\glsinlineemptydescformat` Formats the entry's symbol and location list when the description is empty:

```
7546 \newcommand*{\glsinlineemptydescformat}[2]{}
```

`\glsinlinesubnameformat` Formats the name of the subentry (first argument label, second argument name):

```
7547 \newcommand*{\glsinlinesubnameformat}[2]{\glstarget{#1}{}}
```

`\glsinlinesubdescformat` Formats the subentry's description, symbol and location list:

```
7548 \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.

```
7549 \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.

```
7550 \newglossarystyle{list}{%
```

Use description environment:

```
7551 \renewenvironment{theglossary}{%
```

```
7552 \begin{description}}{\end{description}}%
```

No header at the start of the environment:

```
7553 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7554 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries start a new item in the list:

```
7555 \renewcommand*{\glossentry}[2]{%
```

```
7556 \item[\glssentryitem{##1}]%
```

```
7557 \glstarget{##1}{\glossentryname{##1}}]
```

```
7558 \glossentrydesc{##1}\glspostdescription\space ##2}%
```

Sub-entries continue on the same line:

```
7559 \renewcommand*{\subglossentry}[3]{%
```

```
7560 \glssubentryitem{##2}]%
```

```
7561 \glstarget{##2}{\strut}]%
```

```
7562 \glossentrydesc{##2}\glspostdescription\space ##3.}%
```

```
7563 % \end{macrocode}
```

```
7564 % Add vertical space between groups:
```

```
7565 \changes{3.03}{2012/09/21}{added check for glsnogroupskip}
```

```
7566 % \begin{macrocode}
```

```
7567 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
```

```
7568 }
```

listgroup The listgroup style is like the list style, but the glossary groups have headings.

```
7569 \newglossarystyle{listgroup}{%
```

Base it on the list style:

```
7570 \setglossarystyle{list}%
```

Each group has a heading:

```
7571 \renewcommand*{\glsgroupheading}[1]{\item[\glsgrouptitle{##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.

```
7572 \newglossarystyle{listhypergroup}{%
```

Base it on the list style:

```
7573 \setglossarystyle{list}%
```

Add navigation links at the start of the environment:

```
7574 \renewcommand*{\glossaryheader}{%
```

```
7575 \item[\glsnavigation]}%
```

Each group has a heading with a hypertarget:

```
7576 \renewcommand*{\glsgroupheading}[1]{%
```

```
7577 \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.

```
7578 \newglossarystyle{altlist}{%
```

Base it on the list style:

```
7579 \setglossarystyle{list}%
```

Main (level 0) entries start a new item in the list with a line break after the entry name:

```
7580 \renewcommand*{\glossentry}[2]{%
```

```
7581 \item[\glsentryitem{##1}%
```

```
7582 \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.

```
7583 \mbox{}\par\nobreak\@afterheading
```

```
7584 \glossentrydesc{##1}\glspostdescription\space ##2}%
```

Sub-entries start a new paragraph:

```
7585 \renewcommand{\subglossentry}[3]{%
```

```
7586 \par
```

```
7587 \glssubentryitem{##2}%
```

```
7588 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space ##3}%
```

```
7589 }
```

altlistgroup The altlistgroup glossary style is like the altlist style, but the glossary groups have headings.

```
7590 \newglossarystyle{altlistgroup}{%
```

Base it on the altlist style:

```
7591 \setglossarystyle{altlist}%
```

Each group has a heading:

```
7592 \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.

```
7593 \newglossarystyle{altlisthypergroup}{%
```

Base it on the `altlist` style:

```
7594 \setglossarystyle{altlist}{%
```

Add navigation links at the start of the environment:

```
7595 \renewcommand*{\glossaryheader}{%
```

```
7596 \item[\glsnavigation]}%
```

Each group has a heading with a `hypertarget`:

```
7597 \renewcommand*{\glsgroupheading}[1]{%
```

```
7598 \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.

```
7599 \newglossarystyle{listdotted}{%
```

Base it on the `list` style:

```
7600 \setglossarystyle{list}{%
```

Each main (level 0) entry starts a new item:

```
7601 \renewcommand*{\glossentry}[2]{%
```

```
7602 \item[\makebox[\glslistdottedwidth][l]{%
```

```
7603 \glsentryitem{##1}%
```

```
7604 \glstarget{##1}{\glossentryname{##1}}%
```

```
7605 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}\glossentrydesc{##1}}%
```

Sub entries have the same format as main entries:

```
7606 \renewcommand*{\subglossentry}[3]{%
```

```
7607 \item[\makebox[\glslistdottedwidth][l]{%
```

```
7608 \glssubentryitem{##2}%
```

```
7609 \glstarget{##2}{\glossentryname{##2}}%
```

```
7610 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}\glossentrydesc{##2}}%
```

```
7611 }
```

`\glslistdottedwidth`

```
7612 \newlength\glslistdottedwidth
```

```
7613 \setlength{\glslistdottedwidth}{.5\hsize}
```

`sublistdotted` This style is similar to the `glostylelistdotted` style, except that the main entries just have the name displayed.

```
7614 \newglossarystyle{sublistdotted}{%
```

Base it on the `listdotted` style:

```
7615 \setglossarystyle{listdotted}{%
```

Main (level 0) entries just display the name:

```
7616 \renewcommand*{\glossentry}[2]{%
7617   \item[\glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}}]}%
7618 }
```

4.4 Glossary Styles using longtable (the glossary-long package)

The glossary styles defined in the package used the longtable environment in the glossary.

```
7619 \ProvidesPackage{glossary-long}[2013/11/14 v4.0 (NLCT)]
```

Requires the package:

```
7620 \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`.)

```
7621 \@ifundefined{glsdescwidth}{%
7622   \newlength\glsdescwidth
7623   \setlength{\glsdescwidth}{0.6\hsize}
7624 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column.

```
7625 \@ifundefined{glspagelistwidth}{%
7626   \newlength\glspagelistwidth
7627   \setlength{\glspagelistwidth}{0.1\hsize}
7628 }{}
```

`long` The long glossary style command which uses the longtable environment:

```
7629 \newglossarystyle{long}{%
```

Use longtable with two columns:

```
7630 \renewenvironment{theglossary}%
7631   {\begin{longtable}\lp{\glsdescwidth}}%
7632   {\end{longtable}}%
```

Do nothing at the start of the environment:

```
7633 \renewcommand*{\glossaryheader}{}%
```

No heading between groups:

```
7634 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries displayed in a row:

```
7635 \renewcommand{\glossentry}[2]{%
7636   \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7637   \glossentrydesc{##1}\glspostdescription\space ##2\tabularnewline
7638   }%
```

Sub entries displayed on the following row without the name:

```
7639 \renewcommand{\subglossentry}[3]{%  
7640     &  
7641     \glssubentryitem{##2}%  
7642     \glstarget{##2}{\strut}\glosentrydesc{##2}\glspostdescription\space  
7643     ##3\tabularnewline  
7644 }
```

Blank row between groups:

```
7645 \renewcommand*{\glsnoglobskip}{\ifglsnogroupskip\else &  
7646 \tabularnewline\fi}%  
7647 }
```

longborder The longborder style is like the above, but with horizontal and vertical lines:

```
7648 \newglossarystyle{longborder}{%
```

Base it on the glostylelong style:

```
7649 \setglossarystyle{long}%
```

Use longtable with two columns with vertical lines between each column:

```
7650 \renewenvironment{theglossary}{%  
7651     \begin{longtable}{|l|p{\glsdescwidth}|}{\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
7652 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%  
7653 }
```

longheader The longheader style is like the long style but with a header:

```
7654 \newglossarystyle{longheader}{%
```

Base it on the glostylelong style:

```
7655 \setglossarystyle{long}%
```

Set the table's header:

```
7656 \renewcommand*{\glossaryheader}{%  
7657     \bfseries \entryname & \bfseries \descriptionname\tabularnewline\endhead}%  
7658 }
```

longheaderborder The longheaderborder style is like the long style but with a header and border:

```
7659 \newglossarystyle{longheaderborder}{%
```

Base it on the glostylelongborder style:

```
7660 \setglossarystyle{longborder}%
```

Set the table's header and add horizontal line to table's foot:

```
7661 \renewcommand*{\glossaryheader}{%  
7662     \hline\bfseries \entryname & \bfseries  
7663     \descriptionname\tabularnewline\hline  
7664     \endhead  
7665     \hline\endfoot}%  
7666 }
```

long3col The long3col style is like long but with 3 columns

```
7667 \newglossarystyle{long3col}{%
```

Use a longtable with 3 columns:

```
7668 \renewenvironment{theglossary}{%
```

```
7669 {\begin{longtable}{lp{\glstdescwidth}p{\glspagelistwidth}}}%
```

```
7670 {\end{longtable}}}%
```

No table header:

```
7671 \renewcommand*{\glossaryheader}{}%
```

No headings between groups:

```
7672 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
7673 \renewcommand{\glossentry}[2]{%
```

```
7674 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
```

```
7675 \glossentrydesc{##1} & ##2\tabularnewline
```

```
7676 }%
```

Sub-entries on a separate row (no name, description in second column, page list in third column):

```
7677 \renewcommand{\subglossentry}[3]{%
```

```
7678 &
```

```
7679 \glssubentryitem{##2}%
```

```
7680 \glstarget{##2}{\strut}\glossentrydesc{##2} &
```

```
7681 ##3\tabularnewline
```

```
7682 }%
```

Blank row between groups:

```
7683 \renewcommand*{\glsgroupskip}{}%
```

```
7684 \ifglsnogroupskip\else & &\tabularnewline\fi}%
```

```
7685 }
```

long3colborder The long3colborder style is like the long3col style but with a border:

```
7686 \newglossarystyle{long3colborder}{%
```

Base it on the glostylelong3col style:

```
7687 \setglossarystyle{long3col}%
```

Use a longtable with 3 columns with vertical lines around them:

```
7688 \renewenvironment{theglossary}{%
```

```
7689 {\begin{longtable}{|lp{\glstdescwidth}|p{\glspagelistwidth}|}}%
```

```
7690 {\end{longtable}}}%
```

Place horizontal lines at the head and foot of the table:

```
7691 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
```

```
7692 }
```

long3colheader The long3colheader style is like long3col but with a header row:

```
7693 \newglossarystyle{long3colheader}{%
```


Base it on the `glostylelong3col` style:

```
7694 \setglossarystyle{long3col}%
```

Set the table's header:

```
7695 \renewcommand*{\glossaryheader}{%  
7696 \bfseries\entryname&\bfseries\descriptionname&  
7697 \bfseries\pagelistname\tabularnewline\endhead}%  
7698 }
```

`long3colheaderborder` The `long3colheaderborder` style is like the above but with a border

```
7699 \newglossarystyle{long3colheaderborder}{%
```

Base it on the `glostylelong3colborder` style:

```
7700 \setglossarystyle{long3colborder}%
```

Set the table's header and add horizontal line at table's foot:

```
7701 \renewcommand*{\glossaryheader}{%  
7702 \hline  
7703 \bfseries\entryname&\bfseries\descriptionname&  
7704 \bfseries\pagelistname\tabularnewline\hline\endhead  
7705 \hline\endfoot}%  
7706 }
```

`long4col` The `long4col` style has four columns where the third column contains the value of the associated symbol key.

```
7707 \newglossarystyle{long4col}{%
```

Use a `longtable` with 4 columns:

```
7708 \renewenvironment{theglossary}%  
7709 {\begin{longtable}{llll}}%  
7710 {\end{longtable}}%
```

No table header:

```
7711 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7712 \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):

```
7713 \renewcommand{\glossentry}[2]{%  
7714 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &  
7715 \glossentrydesc{##1} &  
7716 \glossentrysymbol{##1} &  
7717 ##2\tabularnewline  
7718 }%
```

Sub entries on a single row with no name (description in second column, symbol in third column, page list in last column):

```
7719 \renewcommand{\subglossentry}[3]{%  
7720 &
```

```

7721 \glssubentryitem{##2}%
7722 \glstarget{##2}{\strut}\glosentrydesc{##2} &
7723 \glosentrysymbol{##2} & ##3\tabularnewline
7724 }%

```

Blank row between groups:

```

7725 \renewcommand*{\glsgroupskip}{%
7726 \ifglsgnigroupskip\else & & \tabularnewline\fi}%
7727 }

```

long4colheader The long4colheader style is like long4col but with a header row.

```
7728 \newglossarystyle{long4colheader}{%
```

Base it on the glostylelong4col style:

```
7729 \setglossarystyle{long4col}%
```

Table has a header:

```

7730 \renewcommand*{\glossaryheader}{%
7731 \bfseries\entryname&\bfseries\descriptionname&
7732 \bfseries \symbolname&
7733 \bfseries\pagelistname\tabularnewline\endhead}%
7734 }

```

long4colborder The long4colborder style is like long4col but with a border.

```
7735 \newglossarystyle{long4colborder}{%
```

Base it on the glostylelong4col style:

```
7736 \setglossarystyle{long4col}%
```

Use a longtable with 4 columns surrounded by vertical lines:

```

7737 \renewenvironment{theglossary}%
7738 {\begin{longtable}{|l|l|l|l|}}%
7739 {\end{longtable}}%

```

Add horizontal lines to the head and foot of the table:

```

7740 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
7741 }

```

long4colheaderborder The long4colheaderborder style is like the above but with a border.

```
7742 \newglossarystyle{long4colheaderborder}{%
```

Base it on the glostylelong4col style:

```
7743 \setglossarystyle{long4col}%
```

Use a longtable with 4 columns surrounded by vertical lines:

```

7744 \renewenvironment{theglossary}%
7745 {\begin{longtable}{|l|l|l|l|}}%
7746 {\end{longtable}}%

```

Add table header and horizontal line at the table's foot:

```

7747 \renewcommand*{\glossaryheader}{%
7748 \hline\bfseries\entryname&\bfseries\descriptionname&

```

```

7749 \bfseries \symbolname&
7750 \bfseries\pagelistname\tabularnewline\hline\endhead
7751 \hline\endfoot}%
7752 }

```

altlong4col The altlong4col style is like the long4col style but can have multiline descriptions and page lists.

```

7753 \newglossarystyle{altlong4col}{%
    Base it on the glostylelong4col style:
7754 \setglossarystyle{long4col}%
    Use a longtable with 4 columns where the second and last columns may have
    multiple lines in each row:
7755 \renewenvironment{theglossary}%
7756 {\begin{longtable}{lp{\glsgdescwidth}lp{\glspagelistwidth}}}%
7757 {\end{longtable}}%
7758 }

```

altlong4colheader The altlong4colheader style is like altlong4col but with a header row.

```

7759 \newglossarystyle{altlong4colheader}{%
    Base it on the glostylelong4colheader style:
7760 \setglossarystyle{long4colheader}%
    Use a longtable with 4 columns where the second and last columns may have
    multiple lines in each row:
7761 \renewenvironment{theglossary}%
7762 {\begin{longtable}{lp{\glsgdescwidth}lp{\glspagelistwidth}}}%
7763 {\end{longtable}}%
7764 }

```

altlong4colborder The altlong4colborder style is like altlong4col but with a border.

```

7765 \newglossarystyle{altlong4colborder}{%
    Base it on the glostylelong4colborder style:
7766 \setglossarystyle{long4colborder}%
    Use a longtable with 4 columns where the second and last columns may have
    multiple lines in each row:
7767 \renewenvironment{theglossary}%
7768 {\begin{longtable}{|lp{\glsgdescwidth}|lp{\glspagelistwidth}|}%
7769 {\end{longtable}}%
7770 }

```

altlong4colheaderborder The altlong4colheaderborder style is like the above but with a header as well as a border.

```

7771 \newglossarystyle{altlong4colheaderborder}{%
    Base it on the glostylelong4colheaderborder style:
7772 \setglossarystyle{long4colheaderborder}%

```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```

7773 \renewenvironment{theglossary}%
7774   {\begin{longtable}{|l|p{\glstdescwidth}|l|p{\glspagelistwidth|}}%
7775   {\end{longtable}}%
7776 }

```

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.

```

7777 \ProvidesPackage{glossary-longragged}[2013/11/14 v4.0 (NLCT)]

```

Requires the package:

```

7778 \RequirePackage{array}

```

Requires the package:

```

7779 \RequirePackage{longtable}

```

`\glstdescwidth` This is a length that governs the width of the description column. This may have already been defined.

```

7780 \@ifundefined{glstdescwidth}{%
7781   \newlength\glstdescwidth
7782   \setlength{\glstdescwidth}{0.6\hsize}
7783 }{}

```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

```

7784 \@ifundefined{glspagelistwidth}{%
7785   \newlength\glspagelistwidth
7786   \setlength{\glspagelistwidth}{0.1\hsize}
7787 }{}

```

`longragged` The longragged glossary style is like the long but uses ragged right formatting for the description column.

```

7788 \newglossarystyle{longragged}{%

```

Use longtable with two columns:

```

7789   \renewenvironment{theglossary}%
7790   {\begin{longtable}{l>{\raggedright}p{\glstdescwidth}}}%
7791   {\end{longtable}}%

```

Do nothing at the start of the environment:

```

7792   \renewcommand*{\glossaryheader}{}%

```

No heading between groups:

```

7793   \renewcommand*{\glsgroupheading}[1]{}%

```

Main (level 0) entries displayed in a row:

```

7794 \renewcommand{\glossentry}[2]{%
7795   \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7796   \glossentrydesc{##1}\glspostdescription\space ##2%
7797   \tabularnewline
7798 }%
```

Sub entries displayed on the following row without the name:

```

7799 \renewcommand{\subglossentry}[3]{%
7800   &
7801   \glssubentryitem{##2}%
7802   \glstarget{##2}{\strut}\glossentrydesc{##2}%
7803   \glspostdescription\space ##3%
7804   \tabularnewline
7805 }%
```

Blank row between groups:

```

7806 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & \tabularnewline\fi}%
7807 }
```

longraggedborder The longraggedborder style is like the above, but with horizontal and vertical lines:

```

7808 \newglossarystyle{longraggedborder}{%
```

Base it on the glostylelongragged style:

```

7809 \setglossarystyle{longragged}%
```

Use longtable with two columns with vertical lines between each column:

```

7810 \renewenvironment{theglossary}{%
7811   \begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|}%
7812   {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```

7813 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
7814 }
```

longraggedheader The longraggedheader style is like the longragged style but with a header:

```

7815 \newglossarystyle{longraggedheader}{%
```

Base it on the glostylelongragged style:

```

7816 \setglossarystyle{longragged}%
```

Set the table's header:

```

7817 \renewcommand*{\glossaryheader}{%
7818   \bfseries \entryname & \bfseries \descriptionname
7819   \tabularnewline\endhead}%
7820 }
```

raggedheaderborder The longraggedheaderborder style is like the longragged style but with a header and border:

```

7821 \newglossarystyle{longraggedheaderborder}{%
```

Base it on the `glostylelongraggedborder` style:

```
7822 \setglossarystyle{longraggedborder}%
```

Set the table's header and add horizontal line to table's foot:

```
7823 \renewcommand*{\glossaryheader}{%
7824   \hline\bfseries \entryname & \bfseries \descriptionname
7825   \tabularnewline\hline
7826   \endhead
7827   \hline\endfoot}%
7828 }
```

`longragged3col` The `longragged3col` style is like `longragged` but with 3 columns

```
7829 \newglossarystyle{longragged3col}{%
```

Use a longtable with 3 columns:

```
7830 \renewenvironment{theglossary}%
7831   {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}%
7832     >{\raggedright}p{\glspagelistwidth}}}%
7833   {\end{longtable}}%
```

No table header:

```
7834 \renewcommand*{\glossaryheader}{}%
```

No headings between groups:

```
7835 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
7836 \renewcommand{\glossentry}[2]{%
7837   \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7838   \glossentrydesc{##1} & ##2\tabularnewline
7839   }%
```

Sub-entries on a separate row (no name, description in second column, page list in third column):

```
7840 \renewcommand{\subglossentry}[3]{%
7841   &
7842   \glssubentryitem{##2}%
7843   \glstarget{##2}{\strut}\glossentrydesc{##2} &
7844   ##3\tabularnewline
7845   }%
```

Blank row between groups:

```
7846 \renewcommand*{\glsgroupskip}{%
7847   \ifglsnogroupskip\else & &\tabularnewline\fi}%
7848 }
```

`longragged3colborder` The `longragged3colborder` style is like the `longragged3col` style but with a border:

```
7849 \newglossarystyle{longragged3colborder}{%
```

Base it on the `glostylelongragged3col` style:

```
7850 \setglossarystyle{longragged3col}%
```

Use a `longtable` with 3 columns with vertical lines around them:

```
7851 \renewenvironment{theglossary}%  
7852   {\begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|%  
7853     >{\raggedright}p{\glspagelistwidth}|}%  
7854   {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
7855 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%  
7856 }
```

`longragged3colheader` The `longragged3colheader` style is like `longragged3col` but with a header row:

```
7857 \newglossarystyle{longragged3colheader}{%
```

Base it on the `glostylelongragged3col` style:

```
7858 \setglossarystyle{longragged3col}%
```

Set the table's header:

```
7859 \renewcommand*{\glossaryheader}{%  
7860   \bfseries\entryname&\bfseries\descriptionname&  
7861   \bfseries\pagelistname\tabularnewline\endhead}%  
7862 }
```

`longragged3colheaderborder` The `longragged3colheaderborder` style is like the above but with a border

```
7863 \newglossarystyle{longragged3colheaderborder}{%
```

Base it on the `glostylelongragged3colborder` style:

```
7864 \setglossarystyle{longragged3colborder}%
```

Set the table's header and add horizontal line at table's foot:

```
7865 \renewcommand*{\glossaryheader}{%  
7866   \hline  
7867   \bfseries\entryname&\bfseries\descriptionname&  
7868   \bfseries\pagelistname\tabularnewline\hline\endhead  
7869   \hline\endfoot}%  
7870 }
```

`altrlongragged4col` The `altrlongragged4col` style is like the `altrlong4col` style defined in the package, except that ragged right formatting is used for the description and page list columns.

```
7871 \newglossarystyle{altrlongragged4col}{%
```

Use a `longtable` with 4 columns where the second and last columns may have multiple lines in each row:

```
7872 \renewenvironment{theglossary}%  
7873   {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%  
7874     >{\raggedright}p{\glspagelistwidth}}}%  
7875   {\end{longtable}}%
```

No table header:

```
7876 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7877 \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):

```
7878 \renewcommand{\glossentry}[2]{%
7879   \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7880   \glossentrydesc{##1} & \glossentrydesc{##1} &
7881   ##2\tabularnewline
7882 }%
```

Sub entries on a single row with no name (description in second column, symbol in third column, page list in last column):

```
7883 \renewcommand{\subglossentry}[3]{%
7884   &
7885   \glssubentryitem{##2}%
7886   \glstarget{##2}{\strut}\glossentrydesc{##2} &
7887   \glossentrysymbol{##2} & ##3\tabularnewline
7888 }%
```

Blank row between groups:

```
7889 \renewcommand*{\glsgroupskip}{%
7890   \ifglisnogroupskip\else & & \tabularnewline\fi}%
7891 }
```

`ongragged4colheader` The `altlongragged4colheader` style is like `altlongragged4col` but with a header row.

```
7892 \newglossarystyle{altlongragged4colheader}{%
```

Base it on the `glostylealtlongragged4col` style:

```
7893 \setglossarystyle{altlongragged4col}{%
```

Use a `longtable` with 4 columns where the second and last columns may have multiple lines in each row:

```
7894 \renewenvironment{theglossary}%
7895   {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%
7896     >{\raggedright}p{\glspagelistwidth}}}%
7897   {\end{longtable}}%
```

Table has a header:

```
7898 \renewcommand*{\glossaryheader}{%
7899   \bfseries\entryname&\bfseries\descriptionname&
7900   \bfseries \symbolname&
7901   \bfseries\pagelistname\tabularnewline\endhead}%
7902 }
```

`ongragged4colborder` The `altlongragged4colborder` style is like `altlongragged4col` but with a border.

```
7903 \newglossarystyle{altlongragged4colborder}{%
```


Base it on the `glostylealtlongragged4col` style:

```
7904 \setglossarystyle{altlongragged4col}%
```

Use a `longtable` with 4 columns where the second and last columns may have multiple lines in each row:

```
7905 \renewenvironment{theglossary}%  
7906 {\begin{longtable}{|l|>{\raggedright}p{\glstdescwidth}|l|}%  
7907 >{\raggedright}p{\glspagelistwidth}|}%  
7908 {\end{longtable}}%
```

Add horizontal lines to the head and foot of the table:

```
7909 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%  
7910 }
```

`ged4colheaderborder` The `altlongragged4colheaderborder` style is like the above but with a header as well as a border.

```
7911 \newglossarystyle{altlongragged4colheaderborder}{%
```

Base it on the `glostylealtlongragged4col` style:

```
7912 \setglossarystyle{altlongragged4col}%
```

Use a `longtable` with 4 columns where the second and last columns may have multiple lines in each row:

```
7913 \renewenvironment{theglossary}%  
7914 {\begin{longtable}{|l|>{\raggedright}p{\glstdescwidth}|l|}%  
7915 >{\raggedright}p{\glspagelistwidth}|}%  
7916 {\end{longtable}}%
```

Add table header and horizontal line at the table's foot:

```
7917 \renewcommand*{\glossaryheader}{%  
7918 \hline\bfseries\entryname&\bfseries\descriptionname&  
7919 \bfseries \symbolname&  
7920 \bfseries\pagelistname\tabularnewline\hline\endhead  
7921 \hline\endfoot}%  
7922 }
```

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.

```
7923 \ProvidesPackage{glossary-mcols}[2013/11/14 v4.0 (NLCT)]
```

Required packages:

```
7924 \RequirePackage{multicol}  
7925 \RequirePackage{glossary-tree}
```

`\glsmcols` Define macro in which to store the number of columns. (Defaults to 2.)

```
7926 \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.)

```

7927 \newglossarystyle{mcolindex}{%
7928   \setglossarystyle{index}%
7929   \renewenvironment{theglossary}%
7930     {%
7931       \begin{multicols}{\glsmcols}
7932       \setlength{\parindent}{0pt}%
7933       \setlength{\parskip}{0pt plus 0.3pt}%
7934       \let\item\@idxitem}%
7935     {\end{multicols}}%
7936 }
```

mcolindexgroup As mcolindex but has headings:

```

7937 \newglossarystyle{mcolindexgroup}{%
7938   \setglossarystyle{mcolindex}%
7939   \renewcommand*{\glsgroupeheading}[1]{%
7940     \item\textbf{\glsgrouptitle{##1}}\indexspace}%
7941 }
```

mcolindexhypergroup The mcolindexhypergroup style is like the mcolindexgroup style but has hyper navigation.

```

7942 \newglossarystyle{mcolindexhypergroup}{%
  Base it on the glostylemcolindex style:
7943   \setglossarystyle{mcolindex}%
  Put navigation links to the groups at the start of the glossary:
7944   \renewcommand*{\glossaryheader}{%
7945     \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.
7946   \renewcommand*{\glsgroupeheading}[1]{%
7947     \item\textbf{\glsnavigationhypertarget{##1}}{\glsgrouptitle{##1}}}%
7948     \indexspace}%
7949 }
```

mcoltree Multi-column index style. Same as the tree, but puts the glossary in multiple columns.

```

7950 \newglossarystyle{mcoltree}{%
7951   \setglossarystyle{tree}%
7952   \renewenvironment{theglossary}%
7953     {%
7954       \begin{multicols}{\glsmcols}
7955       \setlength{\parindent}{0pt}%
7956       \setlength{\parskip}{0pt plus 0.3pt}%

```

```

7957 }%
7958 {\end{multicols}}}%
7959 }

```

mcoltreegroup Like the mcoltree style but the glossary groups have headings.

```

7960 \newglossarystyle{mcoltreegroup}{%
    Base it on the glostylemcoltree style:
7961   \setglossarystyle{mcoltree}%
    Each group has a heading (in bold) followed by a vertical gap):
7962   \renewcommand{\glsgroupheading}[1]{\par
7963     \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
7964 }

```

mcoltreehypergroup The mcoltreehypergroup style is like the treegroup style, but has a set of links to the groups at the start of the glossary.

```

7965 \newglossarystyle{mcoltreehypergroup}{%
    Base it on the glostylemcoltree style:
7966   \setglossarystyle{mcoltree}%
    Put navigation links to the groups at the start of the theglossary environment:
7967   \renewcommand*{\glossaryheader}{%
7968     \par\noindent\textbf{\glsnavigation}\par\indexspace}%
    Each group has a heading (in bold with a target) followed by a vertical gap):
7969   \renewcommand*{\glsgroupheading}[1]{%
7970     \par\noindent
7971     \textbf{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}\par
7972     \indexspace}%
7973 }

```

mcoltreenoname Multi-column index style. Same as the treenoname, but puts the glossary in multiple columns.

```

7974 \newglossarystyle{mcoltreenoname}{%
7975   \setglossarystyle{treenoname}%
7976   \renewenvironment{theglossary}%
7977   {%
7978     \begin{multicols}{\glsmcols}
7979     \setlength{\parindent}{0pt}%
7980     \setlength{\parskip}{0pt plus 0.3pt}%
7981   }%
7982   {\end{multicols}}}%
7983 }

```

mcoltreenonamegroup Like the mcoltreenoname style but the glossary groups have headings.

```

7984 \newglossarystyle{mcoltreenonamegroup}{%
    Base it on the glostylemcoltreenoname style:
7985   \setglossarystyle{mcoltreenoname}%

```

Give each group a heading:

```
7986 \renewcommand{\glsgroupheading}[1]{\par
7987 \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
7988 }
```

reentreenonamehypergroup The `mcoltreenonamehypergroup` style is like the `mcoltreenonamegroup` style, but has a set of links to the groups at the start of the glossary.

```
7989 \newglossarystyle{mcoltreenonamehypergroup}{%
```

Base it on the `glostylemcoltreenoname` style:

```
7990 \setglossarystyle{mcoltreenoname}%
```

Put navigation links to the groups at the start of the `theglossary` environment:

```
7991 \renewcommand*{\glossaryheader}{%
7992 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
7993 \renewcommand*{\glsgroupheading}[1]{%
7994 \par\noindent
7995 \textbf{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}\par
7996 \indexspace}%
7997 }
```

mcotalttree Multi-column index style. Same as the `almtree`, but puts the glossary in multiple columns.

```
7998 \newglossarystyle{mcotalttree}{%
7999 \setglossarystyle{almtree}%
8000 \renewenvironment{theglossary}%
8001 {%
8002 \begin{multicols}{\glsmcols}
8003 \def\@gls@prevlevel{-1}%
8004 \mbox{}\par
8005 }%
8006 {\par\end{multicols}}%
8007 }
```

mcotalttreegroup Like the `mcotalttree` style but the glossary groups have headings.

```
8008 \newglossarystyle{mcotalttreegroup}{%
```

Base it on the `glostylemcotalttree` style:

```
8009 \setglossarystyle{mcotalttree}%
```

Give each group a heading.

```
8010 \renewcommand{\glsgroupheading}[1]{\par
8011 \def\@gls@prevlevel{-1}%
8012 \hangindent0pt\relax
8013 \parindent0pt\relax
8014 \textbf{\glsgrouptitle{##1}}\par\indexspace}%
8015 }
```

`colalttreehypergroup` The `colalttreehypergroup` style is like the `colalttreegroup` style, but has a set of links to the groups at the start of the glossary.

```
8016 \newglossarystyle{colalttreehypergroup}{%
```

Base it on the `glostylemcolalttree` style:

```
8017 \setglossarystyle{colalttree}{%
```

Put the navigation links in the header

```
8018 \renewcommand*{\glossaryheader}{%
```

```
8019 \par
```

```
8020 \def\@gls@prevlevel{-1}%
```

```
8021 \hangindent0pt\relax
```

```
8022 \parindent0pt\relax
```

```
8023 \textbf{\glsnavigation}\par\indexspace}%
```

Put a hypertarget at the start of each group

```
8024 \renewcommand*{\glsgroupheading}[1]{%
```

```
8025 \par
```

```
8026 \def\@gls@prevlevel{-1}%
```

```
8027 \hangindent0pt\relax
```

```
8028 \parindent0pt\relax
```

```
8029 \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
```

```
8030 \indexspace}}
```

4.7 Glossary Styles using supertabular environment (`glossary-super` package)

The glossary styles defined in the package use the `supertabular` environment.

```
8031 \ProvidesPackage{glossary-super}[2013/11/14 v4.0 (NLCT)]
```

Requires the package:

```
8032 \RequirePackage{supertabular}
```

`\glsdescwidth` This is a length that governs the width of the description column. This may already have been defined if `has` has been loaded.

```
8033 \@ifundefined{glsdescwidth}{%
```

```
8034 \newlength\glsdescwidth
```

```
8035 \setlength{\glsdescwidth}{0.6\hsize}
```

```
8036 }{}}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined if `has` has been loaded.

```
8037 \@ifundefined{glspagelistwidth}{%
```

```
8038 \newlength\glspagelistwidth
```

```
8039 \setlength{\glspagelistwidth}{0.1\hsize}
```

```
8040 }{}}
```

`super` The `super` glossary style uses the `supertabular` environment (it uses lengths defined in the package.)

```
8041 \newglossarystyle{super}{%
```

Put the glossary in a supertabular environment with two columns and no head or tail:

```
8042 \renewenvironment{theglossary}%
8043   {\tablehead{}\tabletail{}}%
8044   \begin{supertabular}{lp{\glsdescwidth}}%
8045   {\end{supertabular}}%
```

Do nothing at the start of the table:

```
8046 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
8047 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries put in a row (name in first column, description and page list in second column):

```
8048 \renewcommand{\glossentry}[2]{%
8049   \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
8050   \glossentrydesc{##1}\glspostdescription\space ##2\tabularnewline
8051 }%
```

Sub entries put in a row (no name, description and page list in second column):

```
8052 \renewcommand{\subglossentry}[3]{%
8053   &
8054   \glssubentryitem{##2}%
8055   \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space
8056   ##3\tabularnewline
8057 }%
```

Blank row between groups:

```
8058 \renewcommand*{\glsgroupskip}{}%
8059 \ifglsnogroupskip\else & \tabularnewline\fi}%
8060 }
```

superborder The superborder style is like the above, but with horizontal and vertical lines:

```
8061 \newglossarystyle{superborder}{}%
```

Base it on the glostylesuper style:

```
8062 \setglossarystyle{super}{}%
```

Put the glossary in a supertabular environment with two columns and a horizontal line in the head and tail:

```
8063 \renewenvironment{theglossary}%
8064   {\tablehead{\hline}\tabletail{\hline}%
8065   \begin{supertabular}{|lp{\glsdescwidth}|}%
8066   {\end{supertabular}}%
8067 }
```

superheader The superheader style is like the super style, but with a header:

```
8068 \newglossarystyle{superheader}{}%
```

Base it on the glostylesuper style:

```
8069 \setglossarystyle{super}{}%
```

Put the glossary in a supertabular environment with two columns, a header and no tail:

```
8070 \renewenvironment{theglossary}%
8071 {\tablehead{\bfseries \entryname &
8072 \bfseries \descriptionname \tabularnewline}%
8073 \tabletail{}}%
8074 \begin{supertabular}{lp{\glstdescwidth}}%
8075 {\end{supertabular}}%
8076 }
```

superheaderborder The superheaderborder style is like the super style but with a header and border:

```
8077 \newglossarystyle{superheaderborder}{%
```

Base it on the glostylesuper style:

```
8078 \setglossarystyle{super}%
```

Put the glossary in a supertabular environment with two columns, a header and horizontal lines above and below the table:

```
8079 \renewenvironment{theglossary}%
8080 {\tablehead{\hline\bfseries \entryname &
8081 \bfseries \descriptionname \tabularnewline\hline}%
8082 \tabletail{\hline}
8083 \begin{supertabular}{|lp{\glstdescwidth}|}%
8084 {\end{supertabular}}%
8085 }
```

super3col The super3col style is like the super style, but with 3 columns:

```
8086 \newglossarystyle{super3col}{%
```

Put the glossary in a supertabular environment with three columns and no head or tail:

```
8087 \renewenvironment{theglossary}%
8088 {\tablehead{}\tabletail{}}%
8089 \begin{supertabular}{lp{\glstdescwidth}p{\glspagelistwidth}}%
8090 {\end{supertabular}}%
```

Do nothing at the start of the table:

```
8091 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
8092 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
8093 \renewcommand{\glossentry}[2]{%
8094 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
8095 \glossentrydesc{##1} & ##2 \tabularnewline
8096 }
```

Sub entries on a row (no name, description in second column, page list in last column):

```
8097 \renewcommand{\subglossentry}[3]{%
8098     &
8099     \glssubentryitem{##2}%
8100     \glstarget{##2}{\strut}\glossentrydesc{##2} &
8101     ##3\tabularnewline
8102 }
```

Blank row between groups:

```
8103 \renewcommand*{\glsgroupskip}{%
8104     \ifglsgnোগroupskip\else & &\tabularnewline\fi}%
8105 }
```

super3colborder The super3colborder style is like the super3col style, but with a border:

```
8106 \newglossarystyle{super3colborder}{%
    Base it on the glostylesuper3col style:
8107     \setglossarystyle{super3col}%
    Put the glossary in a supertabular environment with three columns and a horizontal line in the head and tail:
8108     \renewenvironment{theglossary}%
8109         {\tablehead{\hline}\tabletail{\hline}%
8110         \begin{supertabular}{|l|p{\glsgdescwidth}|p{\glspagelistwidth}|}%
8111         {\end{supertabular}}%
8112 }
```

super3colheader The super3colheader style is like the super3col style but with a header row:

```
8113 \newglossarystyle{super3colheader}{%
    Base it on the glostylesuper3col style:
8114     \setglossarystyle{super3col}%
    Put the glossary in a supertabular environment with three columns, a header and no tail:
8115     \renewenvironment{theglossary}%
8116         {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
8117                     \bfseries\pagelistname\tabularnewline}\tabletail{}}%
8118         \begin{supertabular}{lp{\glsgdescwidth}p{\glspagelistwidth}}%
8119         {\end{supertabular}}%
8120 }
```

super3colheaderborder The super3colheaderborder style is like the super3col style but with a header and border:

```
8121 \newglossarystyle{super3colheaderborder}{%
    Base it on the glostylesuper3colborder style:
8122     \setglossarystyle{super3colborder}%

```


Put the glossary in a supertabular environment with three columns, a header with horizontal lines and a horizontal line in the tail:

```

8123 \renewenvironment{theglossary}%
8124   {\tablehead{\hline
8125     \bfseries\entryname&\bfseries\descriptionname&
8126     \bfseries\pagelistname\tabularnewline\hline}%
8127   \tabletail{\hline}%
8128   \begin{supertabular}{|l|p{\glstdescwidth}|p{\glspagelistwidth|}}%
8129   {\end{supertabular}}%
8130 }
```

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.

```
8131 \newglossarystyle{super4col}{%
```

Put the glossary in a supertabular environment with four columns and no head or tail:

```

8132 \renewenvironment{theglossary}%
8133   {\tablehead{}\tabletail{}}%
8134   \begin{supertabular}{|l|l|l|l|}%
8135   \end{supertabular}}%
```

Do nothing at the start of the table:

```
8136 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
8137 \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:

```

8138 \renewcommand{\glossentry}[2]{%
8139   \glssentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
8140   \glossentrydesc{##1} &
8141   \glossentrysymbol{##1} & ##3\tabularnewline
8142 }%
```

Sub entries on a row with no name, the description in the second column, symbol in third column and page list in last column:

```

8143 \renewcommand{\subglossentry}[3]{%
8144   &
8145   \glssubentryitem{##2}%
8146   \glstarget{##2}{\strut}\glossentrydesc{##2} &
8147   \glossentrysymbol{##2} & ##3\tabularnewline
8148 }%
```

Blank row between groups:

```

8149 \renewcommand*{\glsgroupskip}{%
8150   \ifglsgroupskip\else & & \tabularnewline\fi}%
8151 }
```

super4colheader The super4colheader style is like the super4col but with a header row.

```
8152 \newglossarystyle{super4colheader}{%  
    Base it on the glostylesuper4col style:  
8153   \setglossarystyle{super4col}%  
    Put the glossary in a supertabular environment with four columns, a header and  
    no tail:  
8154   \renewenvironment{theglossary}%  
8155     {\tablehead{\bfseries\entryname&\bfseries\descriptionname&  
8156       \bfseries\symbolname &  
8157       \bfseries\pagelistname\tabularnewline}%  
8158     \tabletail{}}%  
8159     \begin{supertabular}{|l|l|l|l|}%  
8160     {\end{supertabular}}%  
8161 }
```

super4colborder The super4colborder style is like the super4col but with a border.

```
8162 \newglossarystyle{super4colborder}{%  
    Base it on the glostylesuper4col style:  
8163   \setglossarystyle{super4col}%  
    Put the glossary in a supertabular environment with four columns and a hori-  
    zontal line in the head and tail:  
8164   \renewenvironment{theglossary}%  
8165     {\tablehead{\hline}\tabletail{\hline}%  
8166     \begin{supertabular}{|l|l|l|l|}%  
8167     {\end{supertabular}}%  
8168 }
```

super4colheaderborder The super4colheaderborder style is like the super4col but with a header and border.

```
8169 \newglossarystyle{super4colheaderborder}{%  
    Base it on the glostylesuper4col style:  
8170   \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:  
8171   \renewenvironment{theglossary}%  
8172     {\tablehead{\hline\bfseries\entryname&\bfseries\descriptionname&  
8173       \bfseries\symbolname &  
8174       \bfseries\pagelistname\tabularnewline\hline}%  
8175     \tabletail{\hline}%  
8176     \begin{supertabular}{|l|l|l|l|}%  
8177     {\end{supertabular}}%  
8178 }
```

altsuper4col The altsuper4col glossary style is like super4col but has provision for multiline descriptions.

```
8179 \newglossarystyle{altsuper4col}{%  
    Base it on the glostylesuper4col style:  
8180 \setglossarystyle{super4col}%  
    Put the glossary in a supertabular environment with four columns and no head  
    or tail:  
8181 \renewenvironment{theglossary}%  
8182 {\tablehead{}\tabletail{}}%  
8183 \begin{supertabular}{lp{\glsgdescwidth}lp{\glspagelistwidth}}}%  
8184 {\end{supertabular}}}%  
8185 }
```

altsuper4colheader The altsuper4colheader style is like the altsuper4col but with a header row.

```
8186 \newglossarystyle{altsuper4colheader}{%  
    Base it on the glostylesuper4colheader style:  
8187 \setglossarystyle{super4colheader}%  
    Put the glossary in a supertabular environment with four columns, a header and  
    no tail:  
8188 \renewenvironment{theglossary}%  
8189 {\tablehead{\bfseries\entryname&\bfseries\descriptionname&  
8190 \bfseries\symbolname &  
8191 \bfseries\pagelistname\tabularnewline}\tabletail{}}%  
8192 \begin{supertabular}{lp{\glsgdescwidth}lp{\glspagelistwidth}}}%  
8193 {\end{supertabular}}}%  
8194 }
```

altsuper4colborder The altsuper4colborder style is like the altsuper4col but with a border.

```
8195 \newglossarystyle{altsuper4colborder}{%  
    Base it on the glostylesuper4colborder style:  
8196 \setglossarystyle{super4colborder}%  
    Put the glossary in a supertabular environment with four columns and a hori-  
    zontal line in the head and tail:  
8197 \renewenvironment{theglossary}%  
8198 {\tablehead{\hline}\tabletail{\hline}%  
8199 \begin{supertabular}%  
8200 {\lllp{\glsgdescwidth}lllp{\glspagelistwidth}}}%  
8201 {\end{supertabular}}}%  
8202 }
```

per4colheaderborder The altsuper4colheaderborder style is like the altsuper4col but with a header and border.

```
8203 \newglossarystyle{altsuper4colheaderborder}{%
```

Base it on the `glostylessuper4colheaderborder` style:

```
8204 \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:

```
8205 \renewenvironment{theglossary}%
8206   {\tablehead{\hline
8207     \bfseries\entryname &
8208     \bfseries\descriptionname &
8209     \bfseries\symbolname &
8210     \bfseries\pagelistname\tabularnewline\hline}%
8211   \tabletail{\hline}%
8212   \begin{supertabular}%
8213     {\lllp{\glstdescwidth}\lllp{\glspagelistwidth}}}%
8214   {\end{supertabular}}%
8215 }
```

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.

```
8216 \ProvidesPackage{glossary-superragged}[2013/11/14 v4.0 (NLCT)]
```

Requires the package:

```
8217 \RequirePackage{array}
```

Requires the package:

```
8218 \RequirePackage{supertabular}
```

`\glstdescwidth` This is a length that governs the width of the description column. This may already have been defined.

```
8219 \@ifundefined{glstdescwidth}{%
8220   \newlength\glstdescwidth
8221   \setlength{\glstdescwidth}{0.6\hsize}
8222 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

```
8223 \@ifundefined{glspagelistwidth}{%
8224   \newlength\glspagelistwidth
8225   \setlength{\glspagelistwidth}{0.1\hsize}
8226 }{}
```

`superragged` The `superragged` glossary style uses the `supertabular` environment.

```
8227 \newglossarystyle{superragged}{%
```

Put the glossary in a supertabular environment with two columns and no head or tail:

```
8228 \renewenvironment{theglossary}%
8229   {\tablehead{}\tabletail{}}%
8230   \begin{supertabular}{1>{\raggedright}p{\glsgdescwidth}}}%
8231   {\end{supertabular}}%
```

Do nothing at the start of the table:

```
8232 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
8233 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries put in a row (name in first column, description and page list in second column):

```
8234 \renewcommand{\glossentry}[2]{%
8235   \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
8236   \glossentrydesc{##1}\glspostdescription\space ##2%
8237   \tabularnewline
8238 }%
```

Sub entries put in a row (no name, description and page list in second column):

```
8239 \renewcommand{\subglossentry}[3]{%
8240   &
8241   \glssubentryitem{##2}%
8242   \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space
8243   ##3%
8244   \tabularnewline
8245 }%
```

Blank row between groups:

```
8246 \renewcommand*{\glsgroupskip}{\ifglsgnigroupskip\else & \tabularnewline\fi}%
8247 }
```

superraggedborder The superraggedborder style is like the above, but with horizontal and vertical lines:

```
8248 \newglossarystyle{superraggedborder}{%
```

Base it on the glostylessuperragged style:

```
8249 \setglossarystyle{superragged}%
```

Put the glossary in a supertabular environment with two columns and a horizontal line in the head and tail:

```
8250 \renewenvironment{theglossary}%
8251   {\tablehead{\hline}\tabletail{\hline}%
8252   \begin{supertabular}{1>{\raggedright}p{\glsgdescwidth}}}%
8253   {\end{supertabular}}%
8254 }
```

superraggedheader The superraggedheader style is like the super style, but with a header:

```
8255 \newglossarystyle{superraggedheader}{%
```

Base it on the `glostylesuperragged` style:

```
8256 \setglossarystyle{superragged}%
```

Put the glossary in a `supertabular` environment with two columns, a header and no tail:

```
8257 \renewenvironment{theglossary}%
8258 {\tablehead{\bfseries \entryname & \bfseries \descriptionname
8259 \tabularnewline}%
8260 \tabletail{}}%
8261 \begin{supertabular}{1>{\raggedright}p{\glsgdescwidth}}}%
8262 {\end{supertabular}}%
8263 }
```

`rraggedheaderborder` The `superraggedheaderborder` style is like the `superragged` style but with a header and border:

```
8264 \newglossarystyle{superraggedheaderborder}{%
```

Base it on the `glostylesuper` style:

```
8265 \setglossarystyle{superragged}%
```

Put the glossary in a `supertabular` environment with two columns, a header and horizontal lines above and below the table:

```
8266 \renewenvironment{theglossary}%
8267 {\tablehead{\hline\bfseries \entryname &
8268 \bfseries \descriptionname\tabularnewline\hline}%
8269 \tabletail{\hline}
8270 \begin{supertabular}{|1|>{\raggedright}p{\glsgdescwidth}|}}%
8271 {\end{supertabular}}%
8272 }
```

`superragged3col` The `superragged3col` style is like the `superragged` style, but with 3 columns:

```
8273 \newglossarystyle{superragged3col}{%
```

Put the glossary in a `supertabular` environment with three columns and no head or tail:

```
8274 \renewenvironment{theglossary}%
8275 {\tablehead{} \tabletail{}}%
8276 \begin{supertabular}{1>{\raggedright}p{\glsgdescwidth}%
8277 >{\raggedright}p{\glspagelistwidth}}}%
8278 {\end{supertabular}}%
```

Do nothing at the start of the table:

```
8279 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
8280 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
8281 \renewcommand{\glossentry}[2]{}%
8282 \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
```

```

8283 \glossentrydesc{##1} &
8284 ##2\tabularnewline
8285 }%

```

Sub entries on a row (no name, description in second column, page list in last column):

```

8286 \renewcommand{\subglossentry}[3]{%
8287 &
8288 \glssubentryitem{##2}%
8289 \glstarget{##2}{\strut}\glossentrydesc{##2} &
8290 ##3\tabularnewline
8291 }%

```

Blank row between groups:

```

8292 \renewcommand*{\glsgroupskip}{\ifglsgnogroupskip\else & &\tabularnewline\fi}%
8293 }

```

superragged3colborder The `superragged3colborder` style is like the `superragged3col` style, but with a border:

```

8294 \newglossarystyle{superragged3colborder}{%

```

Base it on the `glostylesuperragged3col` style:

```

8295 \setglossarystyle{superragged3col}%

```

Put the glossary in a `supertabular` environment with three columns and a horizontal line in the head and tail:

```

8296 \renewenvironment{theglossary}%
8297 {\tablehead{\hline}\tabletail{\hline}%
8298 \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}|%
8299 >{\raggedright}p{\glspagelistwidth}|}%
8300 {\end{supertabular}}%
8301 }

```

superragged3colheader The `superragged3colheader` style is like the `superragged3col` style but with a header row:

```

8302 \newglossarystyle{superragged3colheader}{%

```

Base it on the `glostylesuperragged3col` style:

```

8303 \setglossarystyle{superragged3col}%

```

Put the glossary in a `supertabular` environment with three columns, a header and no tail:

```

8304 \renewenvironment{theglossary}%
8305 {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
8306 \bfseries\pagelistname\tabularnewline}\tabletail{}}%
8307 \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}%
8308 >{\raggedright}p{\glspagelistwidth}}}%
8309 {\end{supertabular}}%
8310 }

```

ght3colheaderborder The superragged3colheaderborder style is like the superragged3col style but with a header and border:

```
8311 \newglossarystyle{superragged3colheaderborder}{%
```

Base it on the glostylesuperragged3colborder style:

```
8312 \setglossarystyle{superragged3colborder}{%
```

Put the glossary in a supertabular environment with three columns, a header with horizontal lines and a horizontal line in the tail:

```
8313 \renewenvironment{theglossary}{%
8314   {\tablehead{\hline
8315     \bfseries\entryname&\bfseries\descriptionname&
8316     \bfseries\pagelistname\tabularnewline\hline}%
8317   \tabletail{\hline}%
8318   \begin{supertabular}{|l|>{\raggedright}p{\glsdescwidth}|%
8319     >{\raggedright}p{\glspagelistwidth}|}%
8320   {\end{supertabular}}}%
8321 }
```

altsuperragged4col The altsuperragged4col glossary style is like altsuper4col style in the package but uses ragged right formatting in the description and page list columns.

```
8322 \newglossarystyle{altsuperragged4col}{%
```

Put the glossary in a supertabular environment with four columns and no head or tail:

```
8323 \renewenvironment{theglossary}{%
8324   {\tablehead{}\tabletail}%
8325   \begin{supertabular}{|l>{\raggedright}p{\glsdescwidth}l%
8326     >{\raggedright}p{\glspagelistwidth}}}%
8327   {\end{supertabular}}%
```

Do nothing at the start of the table:

```
8328 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
8329 \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:

```
8330 \renewcommand{\glossentry}[2]{%
8331   \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
8332   \glossentrydesc{##1} &
8333   \glossentrysymbol{##1} & ##2\tabularnewline
8334 }
```

Sub entries on a row with no name, the description in the second column, symbol in third column and page list in last column:

```
8335 \renewcommand{\subglossentry}[3]{%
8336   &
8337   \glssubentryitem{##2}%
8338   \glstarget{##2}{\strut}\glossentrydesc{##2} &
```



```

8339     \glossentrysymbol{##2} & ##3\tabularnewline
8340 }%

```

Blank row between groups:

```

8341 \renewcommand*{\glsgroupskip}{\ifglsgnোগroupskip\else & & \tabularnewline\fi}%
8342 }

```

altperragged4colheader The `altperragged4colheader` style is like the `altperragged4col` style but with a header row.

```

8343 \newglossarystyle{altperragged4colheader}{%

```

Base it on the `glostylealtperragged4col` style:

```

8344 \setglossarystyle{altperragged4col}%

```

Put the glossary in a `supertabular` environment with four columns, a header and no tail:

```

8345 \renewenvironment{theglossary}%
8346 {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
8347 \bfseries\symbolname &
8348 \bfseries\pagelistname\tabularnewline}\tabletail{}}%
8349 \begin{supertabular}{1>{\raggedright}p{\glsgdescwidth}1%
8350 >{\raggedright}p{\glspagelistwidth}}}%
8351 {\end{supertabular}}%
8352 }

```

altperragged4colborder The `altperragged4colborder` style is like the `altperragged4col` style but with a border.

```

8353 \newglossarystyle{altperragged4colborder}{%

```

Base it on the `glostylealtperragged4col` style:

```

8354 \setglossarystyle{altperragged4col}%

```

Put the glossary in a `supertabular` environment with four columns and a horizontal line in the head and tail:

```

8355 \renewenvironment{theglossary}%
8356 {\tablehead{\hline}\tabletail{\hline}%
8357 \begin{supertabular}%
8358 {\|1|>{\raggedright}p{\glsgdescwidth}|1|%
8359 >{\raggedright}p{\glspagelistwidth}|}}%
8360 {\end{supertabular}}%
8361 }

```

altperragged4colheaderborder The `altperragged4colheaderborder` style is like the `altperragged4col` style but with a header and border.

```

8362 \newglossarystyle{altperragged4colheaderborder}{%

```

Base it on the `glostylealtperragged4col` style:

```

8363 \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:

```

8364 \renewenvironment{theglossary}%
8365   {\tablehead{\hline
8366     \bfseries\entryname &
8367     \bfseries\descriptionname &
8368     \bfseries\symbolname &
8369     \bfseries\pagelistname\tabularnewline\hline}%
8370   \tabletail{\hline}%
8371   \begin{supertabular}%
8372     {|||>{\raggedright}p{\glsgdescwidth}|l|}%
8373     >{\raggedright}p{\glspagelistwidth}|}}%
8374   {\end{supertabular}}%
8375 }
```

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.

```
8376 \ProvidesPackage{glossary-tree}[2014/03/06 v4.04 (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.

```
8377 \newglossarystyle{index}{%
```

Set the paragraph indentation and skip and define `\item` to be the same as that used by the index:

```

8378 \renewenvironment{theglossary}%
8379   {\setlength{\parindent}{0pt}%
8380   \setlength{\parskip}{0pt plus 0.3pt}%
8381   \let\item\@idxitem}%
8382   {\par}%

```

Do nothing at the start of the environment:

```
8383 \renewcommand*{\glossaryheader}{}%
```

No group headers:

```
8384 \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.

```

8385 \renewcommand*{\glossentry}[2]{%
8386   \item\glsglentryitem{##1}\textbf{\glstarget{##1}{\glossentryname{##1}}}%
8387   \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
8388   \space \glossentrydesc{##1}\glspostdescription\space ##2%
8389 }%
```

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.

```

8390 \renewcommand{\subglossentry}[3]{%
8391   \ifcase##1\relax
8392     % level 0
8393     \item
8394   \or
8395     % level 1
8396     \subitem
8397     \glssubentryitem{##2}%
8398   \else
8399     % all other levels
8400     \subsubitem
8401   \fi
8402   \textbf{\glstarget{##2}{\glossentryname{##2}}}%
8403   \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
8404   \space\glossentrydesc{##2}\glspostdescription\space ##3%
8405 }%
```

Vertical gap between groups is the same as that used by indices:

```

8406 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}}
```

indexgroup The `indexgroup` style is like the `index` style but has headings.

```

8407 \newglossarystyle{indexgroup}{%
```

Base it on the `glostyleindex` style:

```

8408 \setglossarystyle{index}%
```

Add a heading for each group. This puts the group's title in bold followed by a vertical gap.

```

8409 \renewcommand*{\glsgroupheading}[1]{%
8410   \item\textbf{\glsgrouptitle{##1}}\indexspace}%
8411 }
```

indexhypergroup The `indexhypergroup` style is like the `indexgroup` style but has hyper navigation.

```

8412 \newglossarystyle{indexhypergroup}{%
```

Base it on the `glostyleindex` style:

```

8413 \setglossarystyle{index}%
```

Put navigation links to the groups at the start of the glossary:

```

8414 \renewcommand*{\glossaryheader}{%
8415   \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.

```

8416 \renewcommand*{\glsgroupheading}[1]{%
8417   \item\textbf{\glshnavhypertarget{##1}{\glsgrouptitle{##1}}}%
```

```

8418 \indexspace}%
8419 }

```

tree The tree glossary style is similar in style to the index style, but can have arbitrary levels.

```

8420 \newglossarystyle{tree}{%

```

Set the paragraph indentation and skip:

```

8421 \renewenvironment{theglossary}%
8422 {\setlength{\parindent}{0pt}%
8423 \setlength{\parskip}{0pt plus 0.3pt}}%
8424 {}%

```

Do nothing at the start of the theglossary environment:

```

8425 \renewcommand*{\glossaryheader}{}%

```

No group headings:

```

8426 \renewcommand*{\glsgroupheading}[1]{}%

```

Main (level 0) entries: name in bold, followed by symbol in brackets (if it exists), the description and the page list:

```

8427 \renewcommand{\glossentry}[2]{%
8428 \hangindent0pt\relax
8429 \parindent0pt\relax
8430 \glstentryitem{##1}\textbf{\glstarget{##1}{\glossentryname{##1}}}%
8431 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
8432 \space\glossentrydesc{##1}\glspostdescription\space##2\par
8433 }%

```

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.

```

8434 \renewcommand{\subglossentry}[3]{%
8435 \hangindent##1\glstreeindent\relax
8436 \parindent##1\glstreeindent\relax
8437 \ifnum##1=1\relax
8438 \glssubentryitem{##2}%
8439 \fi
8440 \textbf{\glstarget{##2}{\glossentryname{##2}}}%
8441 \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
8442 \space\glossentrydesc{##2}\glspostdescription\space ##3\par
8443 }%

```

Vertical gap between groups is the same as that used by indices:

```

8444 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}}

```

treegroup Like the tree style but the glossary groups have headings.

```

8445 \newglossarystyle{treegroup}{%

```

Base it on the glostyletree style:

```

8446 \setglossarystyle{tree}%

```

Each group has a heading (in bold) followed by a vertical gap):

```
8447 \renewcommand{\glsgroupheading}[1]{\par
8448 \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
8449 }
```

treehypergroup The treehypergroup style is like the treegroup style, but has a set of links to the groups at the start of the glossary.

```
8450 \newglossarystyle{treehypergroup}{%
```

Base it on the glostyletree style:

```
8451 \setglossarystyle{tree}%
```

Put navigation links to the groups at the start of the theglossary environment:

```
8452 \renewcommand*\glossaryheader}{%
8453 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
8454 \renewcommand*\glsgroupheading}[1]{%
8455 \par\noindent
8456 \textbf{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}\par
8457 \indexspace}%
8458 }
```

\glstreeindent Length governing left indent for each level of the tree style.

```
8459 \newlength\glstreeindent
8460 \setlength{\glstreeindent}{10pt}
```

treenoname The treenoname glossary style is like the tree style, but doesn't print the name or symbol for sub-levels.

```
8461 \newglossarystyle{treenoname}{%
```

Set the paragraph indentation and skip:

```
8462 \renewenvironment{theglossary}%
8463 {\setlength{\parindent}{0pt}%
8464 \setlength{\parskip}{0pt plus 0.3pt}}%
8465 {}%
```

No header:

```
8466 \renewcommand*\glossaryheader}{}%
```

No group headings:

```
8467 \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.

```
8468 \renewcommand{\glossentry}[2]{%
8469 \hangindent0pt\relax
8470 \parindent0pt\relax
8471 \glstentryitem{##1}\textbf{\glstarget{##1}{\glossentryname{##1}}}%
8472 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
8473 \space\glossentrydesc{##1}\glspostdescription\space##2\par
8474 }%
```

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.

```

8475 \renewcommand{\subglossentry}[3]{%
8476   \hangindent##1\glstreeindent\relax
8477   \parindent##1\glstreeindent\relax
8478   \ifnum##1=1\relax
8479     \glssubentryitem{##2}%
8480   \fi
8481   \glstarget{##2}{\strut}%
8482   \glossentrydesc{##2}\glspostdescription\space##3\par
8483 }%
```

Vertical gap between groups is the same as that used by indices:

```

8484 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
8485 }
```

`treenonamegroup` Like the `treenoname` style but the glossary groups have headings.

```

8486 \newglossarystyle{treenonamegroup}{%
```

Base it on the `glostyletreenoname` style:

```

8487 \setglossarystyle{treenoname}%
```

Give each group a heading:

```

8488 \renewcommand{\glsgroupheading}[1]{\par
8489   \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
8490 }
```

`treenonamehypergroup` The `treenonamehypergroup` style is like the `treenonamegroup` style, but has a set of links to the groups at the start of the glossary.

```

8491 \newglossarystyle{treenonamehypergroup}{%
```

Base it on the `glostyletreenoname` style:

```

8492 \setglossarystyle{treenoname}%
```

Put navigation links to the groups at the start of the `theglossary` environment:

```

8493 \renewcommand*{\glossaryheader}{%
8494   \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```

8495 \renewcommand*{\glsgroupheading}[1]{%
8496   \par\noindent
8497   \textbf{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}\par
8498   \indexspace}%
8499 }
```

`\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.

```

8500 \newcommand*{\glsssetwidest}[2][0]{%
8501   \expandafter\def\csname @glswidestname\romannumeral#1\endcsname{%
8502     #2}%
8503 }
```

```

\@glswidestname Initialise \@glswidestname.
8504 \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
        \glsssetwidest.
8505 \newglossarystyle{almtree}{%
        Redefine theglossary environment.
8506 \renewenvironment{theglossary}%
8507     {\def\@gls@prevlevel{-1}%
8508      \mbox{}\par}%
8509     {\par}%

        Set the header and group headers to nothing.
8510 \renewcommand*{\glossaryheader}{}%
8511 \renewcommand*{\glsgroupheading}[1]{}%

        Redefine the way that the level 0 entries are displayed.
8512 \renewcommand{\glossentry}[2]{%
8513     \ifnum\@gls@prevlevel=0\relax
8514     \else

        Find out how big the indentation should be by measuring the widest entry.
8515         \settowidth{\glstreeindent}{\textbf{\@glswidestname\space}}%
8516         \fi

        Set the hangindent and paragraph indent.
8517         \hangindent\glstreeindent
8518         \parindent\glstreeindent

        Put the name to the left of the paragraph block.
8519         \makebox[0pt][r]{\makebox[\glstreeindent][l]{%
8520             \glsentryitem{##1}\textbf{\glstarget{##1}{\glossentryname{##1}}}}}%

        If the symbol is missing, ignore it, otherwise put it in brackets.
8521         \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%

        Do the description followed by the description terminator and location list.
8522         \glossentrydesc{##1}\glspostdescription \space ##2\par

        Set the previous level to 0.
8523         \def\@gls@prevlevel{0}%
8524     }%

        Redefine the way sub-entries are displayed.
8525 \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.
8526         \ifnum##1=1\relax
8527         \glssubentryitem{##2}%
8528         \fi

```

If the level hasn't changed, keep the same settings, otherwise adjust \glstreeindent accordingly.

```
8529 \ifnum\@gls@prevlevel=##1\relax
8530 \else
```

Compute the widest entry for this level, or for level 0 if not defined for this level.

Store in \gls@tmplen

```
8531 \@ifundefined{@glswidestname\romannumeral##1}{%
8532 \settowidth{\gls@tmplen}{\textbf{@glswidestname\space}}{%
8533 \settowidth{\gls@tmplen}{\textbf{%
8534 \csname @glswidestname\romannumeral##1\endcsname\space}}}%
```

Determine if going up or down a level

```
8535 \ifnum\@gls@prevlevel<##1\relax
```

Depth has increased, so add the width of the widest entry to \glstreeindent.

```
8536 \setlength\glstreeindent\gls@tmplen
8537 \addtolength\glstreeindent\parindent
8538 \parindent\glstreeindent
8539 \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.

```
8540 \@ifundefined{@glswidestname\romannumeral\@gls@prevlevel}{%
8541 \settowidth{\glstreeindent}{\textbf{%
8542 @glswidestname\space}}{%
8543 \settowidth{\glstreeindent}{\textbf{%
8544 \csname @glswidestname\romannumeral\@gls@prevlevel
8545 \endcsname\space}}}%
```

Subtract this length from the previous level's paragraph indent and set to \glstreeindent.

```
8546 \addtolength\parindent{-\glstreeindent}%
8547 \setlength\glstreeindent\parindent
8548 \fi
8549 \fi
```

Set the hanging indentation.

```
8550 \hangindent\glstreeindent
```

Put the name to the left of the paragraph block

```
8551 \makebox[0pt][r]{\makebox[\gls@tmplen][l]{%
8552 \textbf{\glstarget{##2}{\glossentryname{##2}}}}}%
```

If the symbol is missing, ignore it, otherwise put it in brackets.

```
8553 \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
```

Do the description followed by the description terminator and location list.

```
8554 \glossentrydesc{##2}\glspostdescription\space ##3\par
```


Set the previous level macro to the current level.

```
8555 \def\@gls@prevlevel{##1}%  
8556 }%
```

Vertical gap between groups is the same as that used by indices:

```
8557 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%  
8558 }
```

alttreegroup Like the alttree style but the glossary groups have headings.

```
8559 \newglossarystyle{alttreegroup}{%
```

Base it on the glostylealttree style:

```
8560 \setglossarystyle{alttree}%
```

Give each group a heading.

```
8561 \renewcommand{\glsgroupheading}[1]{\par  
8562 \def\@gls@prevlevel{-1}%  
8563 \hangindent0pt\relax  
8564 \parindent0pt\relax  
8565 \textbf{\glsgetgrouptitle{##1}}\par\indexspace}%  
8566 }
```

alttreehypergroup The alttreehypergroup style is like the alttreegroup style, but has a set of links to the groups at the start of the glossary.

```
8567 \newglossarystyle{alttreehypergroup}{%
```

Base it on the glostylealttree style:

```
8568 \setglossarystyle{alttree}%
```

Put the navigation links in the header

```
8569 \renewcommand*{\glossaryheader}{%  
8570 \par  
8571 \def\@gls@prevlevel{-1}%  
8572 \hangindent0pt\relax  
8573 \parindent0pt\relax  
8574 \textbf{\glsnavigation}\par\indexspace}%
```

Put a hypertarget at the start of each group

```
8575 \renewcommand*{\glsgroupheading}[1]{%  
8576 \par  
8577 \def\@gls@prevlevel{-1}%  
8578 \hangindent0pt\relax  
8579 \parindent0pt\relax  
8580 \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par  
8581 \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.

```

8582 \NeedsTeXFormat{LaTeX2e}
8583 \ProvidesPackage{glossaries-compatible-207}[2011/04/02 v1.0 (NLCT)]

```

`\GlsAddXdyAttribute` Adds an attribute in old format.

```

8584 \ifglxsindy
8585   \renewcommand*\GlsAddXdyAttribute[1]{%
8586     \edef\@xdyattributes{\@xdyattributes ^^J \string"#1\string"}%
8587     \expandafter\toks@\expandafter{\@xdylocref}%
8588     \edef\@xdylocref{\the\toks@ ^^J%
8589       (markup-locref
8590       :open \string"\string~n\string\setentrycounter
8591         {\noexpand\glscounter}%
8592         \expandafter\string\csname#1\endcsname
8593         \expandafter\@gobble\string\{\string" ^^J
8594       :close \string"\expandafter\@gobble\string\}\string" ^^J
8595       :attr \string"#1\string"))}

```

Only has an effect before `\writeist`:

```

8596 \fi

```

`\GlsAddXdyCounters`

```

8597 \renewcommand*\GlsAddXdyCounters[1]{%
8598   \GlossariesWarning{\string\GlsAddXdyCounters\space not available
8599     in compatibility mode.}%
8600 }

```

Add predefined attributes

```

8601 \GlsAddXdyAttribute{glnumberformat}
8602 \GlsAddXdyAttribute{textrm}
8603 \GlsAddXdyAttribute{textsf}
8604 \GlsAddXdyAttribute{texttt}
8605 \GlsAddXdyAttribute{textbf}
8606 \GlsAddXdyAttribute{textmd}
8607 \GlsAddXdyAttribute{textit}
8608 \GlsAddXdyAttribute{textup}
8609 \GlsAddXdyAttribute{textsl}
8610 \GlsAddXdyAttribute{textsc}
8611 \GlsAddXdyAttribute{emph}
8612 \GlsAddXdyAttribute{glshypernumber}
8613 \GlsAddXdyAttribute{hyperrrm}
8614 \GlsAddXdyAttribute{hypersf}
8615 \GlsAddXdyAttribute{hypertt}
8616 \GlsAddXdyAttribute{hyperbf}
8617 \GlsAddXdyAttribute{hypermd}
8618 \GlsAddXdyAttribute{hyperit}
8619 \GlsAddXdyAttribute{hyperup}
8620 \GlsAddXdyAttribute{hypersl}
8621 \GlsAddXdyAttribute{hypersc}
8622 \GlsAddXdyAttribute{hyperemph}

```

\GlsAddXdyLocation Restore v2.07 definition:

```
8623 \ifglxsindy
8624   \renewcommand*{\GlsAddXdyLocation}[2]{%
8625     \edef\xdyuserlocationdefs{%
8626       \xdyuserlocationdefs ^^J%
8627       (define-location-class \string"#1\string"^^J\space\space
8628       \space(#2))
8629     }%
8630     \edef\xdyuserlocationnames{%
8631       \xdyuserlocationnames^^J\space\space\space
8632       \string"#1\string"}%
8633   }
8634 \fi
```

\@do@wrglossary

```
8635 \renewcommand{\@do@wrglossary}[1]{%
  Determine whether to use xindy or makeindex syntax
8636 \ifglxsindy
  Need to determine if the formatting information starts with a ( or ) indicating a
  range.
8637   \expandafter\@glo@check@mkidxrangechar\@glsnumberformat\@nil
8638   \def\@glo@range{}%
8639   \expandafter\if\@glo@prefix(\relax
8640     \def\@glo@range{:open-range}%
8641   \else
8642     \expandafter\if\@glo@prefix)\relax
8643     \def\@glo@range{:close-range}%
8644   \fi
8645   \fi

  Get the location and escape any special characters
8646   \protected@edef\@glslocref{\theglsentrycounter}%
8647   \@gls@checkmkidxchars\@glslocref

  Write to the glossary file using xindy syntax.
8648   \glossary[\csname glo@#1@type\endcsname]{%
8649     (indexentry :tkey (\csname glo@#1@index\endcsname)
8650       :locref \string"\@glslocref\string" %
8651       :attr \string"\@glo@suffix\string" \@glo@range
8652     )
8653   }%
8654 \else

  Convert the format information into the format required for makeindex
8655   \@set@glo@numformat\@glo@numfmt\@gls@counter\@glsnumberformat

  Write to the glossary file using makeindex syntax.
8656   \glossary[\csname glo@#1@type\endcsname]{%
8657     \string\glossaryentry{\csname glo@#1@index\endcsname
```

```

8658 \@gls@encapchar\@glo@numfmt}{\theglsentrycounter}}}%
8659 \fi
8660 }

```

\set@glo@numformat Only had 3 arguments in v2.07

```

8661 \def\set@glo@numformat#1#2#3{%
8662 \expandafter\@glo@check@mkidxrangechar#3\@nil
8663 \protected@edef#1{%
8664 \@glo@prefix setentrycounter[] {#2}%
8665 \expandafter\string\csname\@glo@suffix\endcsname
8666 }%
8667 \@gls@checkmkidxchars#1%
8668 }

```

\writeist Redefine \writeist back to the way it was in v2.07, but change \istfile to \glswrite.

```

8669 \ifglxindy
8670 \def\writeist{%
8671 \openout\glswrite=\istfilename
8672 \write\glswrite{;; xindy style file created by the glossaries
8673 package in compatible-2.07 mode}%
8674 \write\glswrite{;; for document '\jobname' on
8675 \the\year-\the\month-\the\day}%
8676 \write\glswrite{^^J; required styles^^J}
8677 \@for\@xdystyle:=\@xdyrequiredstyles\do{%
8678 \ifx\@xdystyle\@empty
8679 \else
8680 \protected@write\glswrite{{(require
8681 \string"\@xdystyle.xdy\string")}}%
8682 \fi
8683 }%
8684 \write\glswrite{^^J%
8685 ; list of allowed attributes (number formats)^^J}%
8686 \write\glswrite{(define-attributes ((\@xdyattributes)))}%
8687 \write\glswrite{^^J; user defined alphabets^^J}%
8688 \write\glswrite{\@xdyuseralphabets}%
8689 \write\glswrite{^^J; location class definitions^^J}%
8690 \protected@edef\@gls@roman{\@roman{0}\string"
8691 \string"roman-numbers-lowercase\string" :sep \string"}}%
8692 \@onelevel@sanitize\@gls@roman
8693 \edef\@tmp{\string" \string"roman-numbers-lowercase\string"
8694 :sep \string"}%
8695 \@onelevel@sanitize\@tmp
8696 \ifx\@tmp\@gls@roman
8697 \write\glswrite{(define-location-class
8698 \string"roman-page-numbers\string"^^J\space\space\space
8699 (\string"roman-numbers-lowercase\string")
8700 :min-range-length \@glsminrange)}}%
8701 \else

```

```

8702     \write\glswrite{(define-location-class
8703       \string"roman-page-numbers\string"^^J\space\space\space
8704       (:sep "\@gls@roman")
8705       :min-range-length \@glsminrange)}}%
8706   \fi
8707   \write\glswrite{(define-location-class
8708     \string"Roman-page-numbers\string"^^J\space\space\space
8709     (\string"roman-numbers-uppercase\string")
8710     :min-range-length \@glsminrange)}}%
8711   \write\glswrite{(define-location-class
8712     \string"arabic-page-numbers\string"^^J\space\space\space
8713     (\string"arabic-numbers\string")
8714     :min-range-length \@glsminrange)}}%
8715   \write\glswrite{(define-location-class
8716     \string"alpha-page-numbers\string"^^J\space\space\space
8717     (\string"alpha\string")
8718     :min-range-length \@glsminrange)}}%
8719   \write\glswrite{(define-location-class
8720     \string"Alpha-page-numbers\string"^^J\space\space\space
8721     (\string"ALPHA\string")
8722     :min-range-length \@glsminrange)}}%
8723   \write\glswrite{(define-location-class
8724     \string"Appendix-page-numbers\string"^^J\space\space\space
8725     (\string"ALPHA\string"
8726     :sep \string"\@glsAlphacompositor\string"
8727     \string"arabic-numbers\string")
8728     :min-range-length \@glsminrange)}}%
8729   \write\glswrite{(define-location-class
8730     \string"arabic-section-numbers\string"^^J\space\space\space
8731     (\string"arabic-numbers\string"
8732     :sep \string"\glscompositor\string"
8733     \string"arabic-numbers\string")
8734     :min-range-length \@glsminrange)}}%
8735   \write\glswrite{^^J; user defined location classes}%
8736   \write\glswrite{\@xdyuserlocationdefs}%
8737   \write\glswrite{^^J; define cross-reference class^^J}%
8738   \write\glswrite{(define-crossref-class \string"see\string"
8739     :unverified )}%
8740   \write\glswrite{(markup-crossref-list
8741     :class \string"see\string"^^J\space\space\space
8742     :open \string"\string\glsseeformat\string"
8743     :close \string"{}\string")}%
8744   \write\glswrite{^^J; define the order of the location classes}%
8745   \write\glswrite{(define-location-class-order
8746     (\@xdylocationclassorder))}%
8747   \write\glswrite{^^J; define the glossary markup^^J}%
8748   \write\glswrite{(markup-index^^J\space\space\space
8749     :open \string"\string
8750     \glossarysection[\string\glossarytoctitle]{\string

```

```

8751 \glossarytitle}\string\glossarypreamble\string~n\string\begin
8752 {theglossary}\string\glossaryheader\string~n\string" ^^J\space
8753 \space\space:close \string"\expandafter\@gobble
8754 \string%\string~n\string
8755 \end{theglossary}\string\glossarypostamble
8756 \string~n\string" ^^J\space\space\space
8757 :tree}}%
8758 \write\glswrite{(markup-letter-group-list
8759 :sep \string"\string\glsgroupskip\string~n\string"))}%
8760 \write\glswrite{(markup-indexentry
8761 :open \string"\string\relax \string\glsresetentrylist
8762 \string~n\string"))}%
8763 \write\glswrite{(markup-locclass-list :open
8764 \string"\glsopenbrace\string\glossaryentrynumbers
8765 \glsopenbrace\string\relax\space \string"^^J\space\space\space
8766 :sep \string", \string"
8767 :close \string"\glsclosebrace\glsclosebrace\string"))}%
8768 \write\glswrite{(markup-locref-list
8769 :sep \string"\string\delimN\space\string"))}%
8770 \write\glswrite{(markup-range
8771 :sep \string"\string\delimR\space\string"))}%
8772 \@onelevel@sanitize\gls@suffixF
8773 \@onelevel@sanitize\gls@suffixFF
8774 \ifx\gls@suffixF\@empty
8775 \else
8776 \write\glswrite{(markup-range
8777 :close "\gls@suffixF" :length 1 :ignore-end))}%
8778 \fi
8779 \ifx\gls@suffixFF\@empty
8780 \else
8781 \write\glswrite{(markup-range
8782 :close "\gls@suffixFF" :length 2 :ignore-end))}%
8783 \fi
8784 \write\glswrite{^^J; define format to use for locations^^J}%
8785 \write\glswrite{\@xdylocref}%
8786 \write\glswrite{^^J; define letter group list format^^J}%
8787 \write\glswrite{(markup-letter-group-list
8788 :sep \string"\string\glsgroupskip\string~n\string"))}%
8789 \write\glswrite{^^J; letter group headings^^J}%
8790 \write\glswrite{(markup-letter-group
8791 :open-head \string"\string\glsgroupheading
8792 \glsopenbrace\string"^^J\space\space\space
8793 :close-head \string"\glsclosebrace\string"))}%
8794 \write\glswrite{^^J; additional letter groups^^J}%
8795 \write\glswrite{\@xdylettergroups}%
8796 \write\glswrite{^^J; additional sort rules^^J}
8797 \write\glswrite{\@xdysortrules}%
8798 \noist}
8799 \else

```

```

8800 \edef\@gls@actualchar{\string?}
8801 \edef\@gls@encapchar{\string|}
8802 \edef\@gls@levelchar{\string!}
8803 \edef\@gls@quotechar{\string"}
8804 \def\writeist{\relax
8805   \openout\glswrite=\istfilename
8806   \write\glswrite{\expandafter\@gobble\string\% makeindex style file
8807     created by the glossaries package}
8808   \write\glswrite{\expandafter\@gobble\string\% for document
8809     '\jobname' on \the\year-\the\month-\the\day}
8810   \write\glswrite{actual '\@gls@actualchar'}
8811   \write\glswrite{encap '\@gls@encapchar'}
8812   \write\glswrite{level '\@gls@levelchar'}
8813   \write\glswrite{quote '\@gls@quotechar'}
8814   \write\glswrite{keyword \string"\string\glossaryentry\string"}
8815   \write\glswrite{preamble \string"\string\glossarysection[\string
8816     \glossarytoctitle]{\string\glossarytitle}\string
8817     \glossarypreamble\string\n\string\begin{theglossary}\string
8818     \glossaryheader\string\n\string"}
8819   \write\glswrite{postamble \string"\string%\string\n\string
8820     \end{theglossary}\string\glossarypostamble\string\n
8821     \string"}
8822   \write\glswrite{group_skip \string"\string\glsgroupskip\string\n
8823     \string"}
8824   \write\glswrite{item_0 \string"\string%\string\n\string"}
8825   \write\glswrite{item_1 \string"\string%\string\n\string"}
8826   \write\glswrite{item_2 \string"\string%\string\n\string"}
8827   \write\glswrite{item_01 \string"\string%\string\n\string"}
8828   \write\glswrite{item_x1
8829     \string"\string\relax \string\glsresetentrylist\string\n
8830     \string"}
8831   \write\glswrite{item_12 \string"\string%\string\n\string"}
8832   \write\glswrite{item_x2
8833     \string"\string\relax \string\glsresetentrylist\string\n
8834     \string"}
8835   \write\glswrite{delim_0 \string"\string\{\string
8836     \glossaryentrynumbers\string\{\string\relax \string"}
8837   \write\glswrite{delim_1 \string"\string\{\string
8838     \glossaryentrynumbers\string\{\string\relax \string"}
8839   \write\glswrite{delim_2 \string"\string\{\string
8840     \glossaryentrynumbers\string\{\string\relax \string"}
8841   \write\glswrite{delim_t \string"\string\}\string\}\string"}
8842   \write\glswrite{delim_n \string"\string\delimN \string"}
8843   \write\glswrite{delim_r \string"\string\delimR \string"}
8844   \write\glswrite{headings_flag 1}
8845   \write\glswrite{heading_prefix
8846     \string"\string\glsgroupheading\string\{\string"}
8847   \write\glswrite{heading_suffix
8848     \string"\string\}\string\relax

```

```

8849     \string\glsresetentrylist \string"}
8850 \write\glswrite{symhead_positive \string"glssymbols\string"}
8851 \write\glswrite{numhead_positive \string"glsnnumbers\string"}
8852 \write\glswrite{page_compositor \string"\glscompositor\string"}
8853 \@gls@escbsdq\gls@suffixF
8854 \@gls@escbsdq\gls@suffixFF
8855 \ifx\gls@suffixF\@empty
8856 \else
8857     \write\glswrite{suffix_2p \string"\gls@suffixF\string"}
8858 \fi
8859 \ifx\gls@suffixFF\@empty
8860 \else
8861     \write\glswrite{suffix_3p \string"\gls@suffixFF\string"}
8862 \fi
8863 \noist
8864 }
8865 \fi

```

\noist

```
8866 \renewcommand*{\noist}{\let\writeist\relax}
```

Compatibility macros.

```

8867 \NeedsTeXFormat{LaTeX2e}
8868 \ProvidesPackage{glossaries-compatible-307}[2013/11/14 v4.0 (NLCT)]

```

Compatibility macros for predefined glossary styles:

compatglossarystyle Defines a compatibility glossary style.

```

8869 \newcommand{\compatglossarystyle}[2]{%
8870   \ifcsundef{@glscompstyle@#1}%
8871   {%
8872     \csdef{@glscompstyle@#1}{#2}%
8873   }%
8874   {%
8875     \PackageError{glossaries}{Glossary compatibility style ‘#1’ is already defined}{}%
8876   }%
8877 }

```

Backward compatible inline style.

```

8878 \compatglossarystyle{inline}{%
8879   \renewcommand{\glossaryentryfield}[5]{%
8880     \glsinlinedopostchild
8881     \gls@inlinesep
8882     \def\glo@desc{##3}%
8883     \def\@no@post@desc{\nopostdesc}%
8884     \glsentryitem{##1}\glsinlinenameformat{##1}{##2}%
8885     \ifx\glo@desc\@no@post@desc
8886       \glsinlineemptydescformat{##4}{##5}%
8887     \else
8888       \ifstrempy{##3}%

```



```

8889     {\glsinlineemptydescformat{##4}{##5}}%
8890     {\glsinlinedescformat{##3}{##4}{##5}}%
8891     \fi
8892     \ifglshaschildren{##1}%
8893     {%
8894         \glsresetsubentrycounter
8895         \glsinlineparentchildseparator
8896         \def\gls@inlinesubsep{%
8897             \def\gls@inlinepostchild{\glsinlinepostchild}%
8898         }%
8899         {%
8900             \def\gls@inlinesep{\glsinlineseparator}%
8901         }%

```

Sub-entries display description:

```

8902     \renewcommand{\glossarysubentryfield}[6]{%
8903         \gls@inlinesubsep%
8904         \glsinlinesubnameformat{##2}{##3}%
8905         \glssubentryitem{##2}\glsinlinesubdescformat{##4}{##5}{##6}%
8906         \def\gls@inlinesubsep{\glsinlinesubseparator}%
8907     }%
8908 }

```

Backward compatible list style.

```

8909 \compatglossarystyle{list}{%
8910     \renewcommand*\glossaryentryfield[5]{%
8911         \item[\glsentryitem{##1}\glstarget{##1}{##2}]
8912         ##3\glspostdescription\space ##5}%

```

Sub-entries continue on the same line:

```

8913     \renewcommand*\glossarysubentryfield[6]{%
8914         \glssubentryitem{##2}%
8915         \glstarget{##2}{\strut}##4\glspostdescription\space ##6.}%
8916 }

```

Backward compatible listgroup style.

```

8917 \compatglossarystyle{listgroup}{%
8918     \csuse{@glscompstyle@list}%
8919 }%

```

Backward compatible listhypergroup style.

```

8920 \compatglossarystyle{listhypergroup}{%
8921     \csuse{@glscompstyle@list}%
8922 }%

```

Backward compatible altlist style.

```

8923 \compatglossarystyle{altlist}{%
8924     \renewcommand*\glossaryentryfield[5]{%
8925         \item[\glsentryitem{##1}\glstarget{##1}{##2}]%
8926         \mbox{}\par\nobreak\@afterheading
8927         ##3\glspostdescription\space ##5}%
8928     \renewcommand{\glossarysubentryfield}[6]{%

```

```

8929 \par
8930 \glssubentryitem{##2}%
8931 \glstarget{##2}{\strut}##4\glspostdescription\space ##6}%
8932 }%

```

Backward compatible altlistgroup style.

```

8933 \compatglossarystyle{altlistgroup}{%
8934 \csuse{@glscmpstyle@altlist}%
8935 }%

```

Backward compatible altlisthypergroup style.

```

8936 \compatglossarystyle{altlisthypergroup}{%
8937 \csuse{@glscmpstyle@altlist}%
8938 }%

```

Backward compatible listdotted style.

```

8939 \compatglossarystyle{listdotted}{%
8940 \renewcommand*{\glossaryentryfield}[5]{%
8941 \item[]\makebox[\glslistdottedwidth][l]{%
8942 \glssubentryitem{##1}\glstarget{##1}{##2}%
8943 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##3}%
8944 \renewcommand*{\glossarysubentryfield}[6]{%
8945 \item[]\makebox[\glslistdottedwidth][l]{%
8946 \glssubentryitem{##2}%
8947 \glstarget{##2}{##3}%
8948 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##4}%
8949 }%

```

Backward compatible sublistdotted style.

```

8950 \compatglossarystyle{sublistdotted}{%
8951 \csuse{@glscmpstyle@listdotted}%
8952 \renewcommand*{\glossaryentryfield}[5]{%
8953 \item[\glssubentryitem{##1}\glstarget{##1}{##2}]}%
8954 }%

```

Backward compatible long style.

```

8955 \compatglossarystyle{long}{%
8956 \renewcommand*{\glossaryentryfield}[5]{%
8957 \glssubentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\}%
8958 \renewcommand*{\glossarysubentryfield}[6]{%
8959 &
8960 \glssubentryitem{##2}%
8961 \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\}%
8962 }%

```

Backward compatible longborder style.

```

8963 \compatglossarystyle{longborder}{%
8964 \csuse{@glscmpstyle@long}%
8965 }%

```

Backward compatible longheader style.

```

8966 \compatglossarystyle{longheader}{%

```

```
8967 \csuse{@glscompstyle@long}%
8968 }%
```

Backward compatible longheaderborder style.

```
8969 \compatglossarystyle{longheaderborder}{%
8970 \csuse{@glscompstyle@long}%
8971 }%
```

Backward compatible long3col style.

```
8972 \compatglossarystyle{long3col}{%
8973 \renewcommand*{\glossaryentryfield}[5]{%
8974 \glentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\\}%
8975 \renewcommand*{\glossarysubentryfield}[6]{%
8976 &
8977 \glsubentryitem{##2}%
8978 \glstarget{##2}{\strut}##4 & ##6\\}%
8979 }%
```

Backward compatible long3colborder style.

```
8980 \compatglossarystyle{long3colborder}{%
8981 \csuse{@glscompstyle@long3col}%
8982 }%
```

Backward compatible long3colheader style.

```
8983 \compatglossarystyle{long3colheader}{%
8984 \csuse{@glscompstyle@long3col}%
8985 }%
```

Backward compatible long3colheaderborder style.

```
8986 \compatglossarystyle{long3colheaderborder}{%
8987 \csuse{@glscompstyle@long3col}%
8988 }%
```

Backward compatible long4col style.

```
8989 \compatglossarystyle{long4col}{%
8990 \renewcommand*{\glossaryentryfield}[5]{%
8991 \glentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\\}%
8992 \renewcommand*{\glossarysubentryfield}[6]{%
8993 &
8994 \glsubentryitem{##2}%
8995 \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%
8996 }%
```

Backward compatible long4colheader style.

```
8997 \compatglossarystyle{long4colheader}{%
8998 \csuse{@glscompstyle@long4col}%
8999 }%
```

Backward compatible long4colborder style.

```
9000 \compatglossarystyle{long4colborder}{%
9001 \csuse{@glscompstyle@long4col}%
9002 }%
```

Backward compatible long4colheaderborder style.

```
9003 \compatglossarystyle{long4colheaderborder}{%
9004 \csuse{@glscompstyle@long4col}%
9005 }%
```

Backward compatible altlong4col style.

```
9006 \compatglossarystyle{altlong4col}{%
9007 \csuse{@glscompstyle@long4col}%
9008 }%
```

Backward compatible altlong4colheader style.

```
9009 \compatglossarystyle{altlong4colheader}{%
9010 \csuse{@glscompstyle@long4col}%
9011 }%
```

Backward compatible altlong4colborder style.

```
9012 \compatglossarystyle{altlong4colborder}{%
9013 \csuse{@glscompstyle@long4col}%
9014 }%
```

Backward compatible altlong4colheaderborder style.

```
9015 \compatglossarystyle{altlong4colheaderborder}{%
9016 \csuse{@glscompstyle@long4col}%
9017 }%
```

Backward compatible long style.

```
9018 \compatglossarystyle{longragged}{%
9019 \renewcommand*{\glossaryentryfield}[5]{%
9020 \glstarget{##1}{\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%
9021 \tabularnewline}%
9022 \renewcommand*{\glossarysubentryfield}[6]{%
9023 &
9024 \glssubentryitem{##2}%
9025 \glstarget{##2}{\strut}##4\glspostdescription\space ##6%
9026 \tabularnewline}%
9027 }%
```

Backward compatible longraggedborder style.

```
9028 \compatglossarystyle{longraggedborder}{%
9029 \csuse{@glscompstyle@longragged}%
9030 }%
```

Backward compatible longraggedheader style.

```
9031 \compatglossarystyle{longraggedheader}{%
9032 \csuse{@glscompstyle@longragged}%
9033 }%
```

Backward compatible longraggedheaderborder style.

```
9034 \compatglossarystyle{longraggedheaderborder}{%
9035 \csuse{@glscompstyle@longragged}%
9036 }%
```

Backward compatible longragged3col style.

```
9037 \compatglossarystyle{longragged3col}{%
9038   \renewcommand*{\glossaryentryfield}[5]{%
9039     \glstarget{##1}{\glstarget{##1}{##2} & ##3 & ##5\tabularnewline}%
9040   \renewcommand*{\glossarysubentryfield}[6]{%
9041     &
9042     \glssubentryitem{##2}%
9043     \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%
9044 }%
```

Backward compatible longragged3colborder style.

```
9045 \compatglossarystyle{longragged3colborder}{%
9046   \csuse{@glscmpstyle@longragged3col}%
9047 }%
```

Backward compatible longragged3colheader style.

```
9048 \compatglossarystyle{longragged3colheader}{%
9049   \csuse{@glscmpstyle@longragged3col}%
9050 }%
```

Backward compatible longragged3colheaderborder style.

```
9051 \compatglossarystyle{longragged3colheaderborder}{%
9052   \csuse{@glscmpstyle@longragged3col}%
9053 }%
```

Backward compatible altlongragged4col style.

```
9054 \compatglossarystyle{altlongragged4col}{%
9055   \renewcommand*{\glossaryentryfield}[5]{%
9056     \glstarget{##1}{\glstarget{##1}{##2} & ##3 & ##4 & ##5\tabularnewline}%
9057   \renewcommand*{\glossarysubentryfield}[6]{%
9058     &
9059     \glssubentryitem{##2}%
9060     \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%
9061 }%
```

Backward compatible altlongragged4colheader style.

```
9062 \compatglossarystyle{altlongragged4colheader}{%
9063   \csuse{@glscmpstyle@altlong4col}%
9064 }%
```

Backward compatible altlongragged4colborder style.

```
9065 \compatglossarystyle{altlongragged4colborder}{%
9066   \csuse{@glscmpstyle@altlong4col}%
9067 }%
```

Backward compatible altlongragged4colheaderborder style.

```
9068 \compatglossarystyle{altlongragged4colheaderborder}{%
9069   \csuse{@glscmpstyle@altlong4col}%
9070 }%
```

Backward compatible index style.

```
9071 \compatglossarystyle{index}{%
```

```

9072 \renewcommand*\glossaryentryfield}[5]{%
9073   \item\glentryitem{##1}\textbf{\glstarget{##1}{##2}}%
9074   \ifx\relax##4\relax
9075   \else
9076     \space(##4)%
9077   \fi
9078   \space ##3\glspostdescription \space ##5}%
9079 \renewcommand*\glossarysubentryfield}[6]{%
9080   \ifcase##1\relax
9081     % level 0
9082     \item
9083   \or
9084     % level 1
9085     \subitem
9086     \glssubentryitem{##2}%
9087   \else
9088     % all other levels
9089     \subsubitem
9090   \fi
9091   \textbf{\glstarget{##2}{##3}}%
9092   \ifx\relax##5\relax
9093   \else
9094     \space(##5)%
9095   \fi
9096   \space##4\glspostdescription\space ##6}%
9097 }%

```

Backward compatible indexgroup style.

```

9098 \compatglossarystyle{indexgroup}{%
9099   \csuse{@glscmpstyle@index}%
9100 }%

```

Backward compatible indexhypergroup style.

```

9101 \compatglossarystyle{indexhypergroup}{%
9102   \csuse{@glscmpstyle@index}%
9103 }%

```

Backward compatible tree style.

```

9104 \compatglossarystyle{tree}{%
9105   \renewcommand*\glossaryentryfield}[5]{%
9106     \hangindent0pt\relax
9107     \parindent0pt\relax
9108     \glentryitem{##1}\textbf{\glstarget{##1}{##2}}%
9109     \ifx\relax##4\relax
9110     \else
9111       \space(##4)%
9112     \fi
9113     \space ##3\glspostdescription \space ##5\par}%
9114   \renewcommand*\glossarysubentryfield}[6]{%
9115     \hangindent##1\glstreeindent\relax
9116     \parindent##1\glstreeindent\relax

```

```

9117 \ifnum##1=1\relax
9118 \glssubentryitem{##2}%
9119 \fi
9120 \textbf{\glstarget{##2}{##3}}%
9121 \ifx\relax##5\relax
9122 \else
9123 \space(##5)%
9124 \fi
9125 \space##4\glspostdescription\space ##6\par}%
9126 }%

```

Backward compatible treegroup style.

```

9127 \compatglossarystyle{treegroup}{%
9128 \csuse{@glscmpstyle@tree}%
9129 }%

```

Backward compatible treehypergroup style.

```

9130 \compatglossarystyle{treehypergroup}{%
9131 \csuse{@glscmpstyle@tree}%
9132 }%

```

Backward compatible treenoname style.

```

9133 \compatglossarystyle{treenoname}{%
9134 \renewcommand{\glossaryentryfield}[5]{%
9135 \hangindent0pt\relax
9136 \parindent0pt\relax
9137 \glssubentryitem{##1}\textbf{\glstarget{##1}{##2}}%
9138 \ifx\relax##4\relax
9139 \else
9140 \space(##4)%
9141 \fi
9142 \space ##3\glspostdescription \space ##5\par}%
9143 \renewcommand{\glossarysubentryfield}[6]{%
9144 \hangindent##1\glstreeindent\relax
9145 \parindent##1\glstreeindent\relax
9146 \ifnum##1=1\relax
9147 \glssubentryitem{##2}%
9148 \fi
9149 \glstarget{##2}{\strut}%
9150 ##4\glspostdescription\space ##6\par}%
9151 }%

```

Backward compatible treenonamegroup style.

```

9152 \compatglossarystyle{treenonamegroup}{%
9153 \csuse{@glscmpstyle@treenoname}%
9154 }%

```

Backward compatible treenonamehypergroup style.

```

9155 \compatglossarystyle{treenonamehypergroup}{%
9156 \csuse{@glscmpstyle@treenoname}%
9157 }%

```

Backward compatible alttree style.

```

9158 \compatglossarystyle{alttree}{%
9159   \renewcommand{\glossaryentryfield}[5]{%
9160     \ifnum\@gls@prevlevel=0\relax
9161     \else
9162       \settowidth{\glstreeindent}{\textbf{\@glswidestname\space}}%
9163       \hangindent\glstreeindent
9164       \parindent\glstreeindent
9165     \fi
9166     \makebox[0pt][r]{\makebox[\glstreeindent][l]{%
9167       \glstryitem{##1}\textbf{\glstarget{##1}{##2}}}%
9168     \ifx\relax##4\relax
9169     \else
9170       (##4)\space
9171     \fi
9172     ##3\glspostdescription \space ##5\par
9173     \def\@gls@prevlevel{0}%
9174   }%
9175   \renewcommand{\glossarysubentryfield}[6]{%
9176     \ifnum##1=1\relax
9177       \glssubentryitem{##2}%
9178     \fi
9179     \ifnum\@gls@prevlevel=##1\relax
9180     \else
9181       \@ifundefined{\glswidestname\romannumeral##1}{%
9182         \settowidth{\gls@tmplen}{\textbf{\@glswidestname\space}}{%
9183         \settowidth{\gls@tmplen}{\textbf{%
9184           \csname @glswidestname\romannumeral##1\endcsname\space}}}%
9185       \ifnum\@gls@prevlevel<##1\relax
9186         \setlength\glstreeindent\gls@tmplen
9187         \addtolength\glstreeindent\parindent
9188         \parindent\glstreeindent
9189       \else
9190         \@ifundefined{\glswidestname\romannumeral\@gls@prevlevel}{%
9191           \settowidth{\glstreeindent}{\textbf{%
9192             \@glswidestname\space}}{%
9193           \settowidth{\glstreeindent}{\textbf{%
9194             \csname @glswidestname\romannumeral\@gls@prevlevel
9195             \endcsname\space}}}%
9196           \addtolength\parindent{-\glstreeindent}%
9197           \setlength\glstreeindent\parindent
9198         \fi
9199       \fi
9200       \hangindent\glstreeindent
9201       \makebox[0pt][r]{\makebox[\gls@tmplen][l]{%
9202         \textbf{\glstarget{##2}{##3}}}%
9203       \ifx##5\relax\relax
9204       \else
9205         (##5)\space

```



```

9206 \fi
9207 ##4\glspostdescription\space ##6\par
9208 \def\@gls@prevlevel{##1}%
9209 }%
9210 }%

```

Backward compatible alttreegroup style.

```

9211 \compatglossarystyle{alttreegroup}{%
9212 \csuse{@glscompstyle@almtree}%
9213 }%

```

Backward compatible alttreehypergroup style.

```

9214 \compatglossarystyle{alttreehypergroup}{%
9215 \csuse{@glscompstyle@almtree}%
9216 }%

```

Backward compatible mcolindex style.

```

9217 \compatglossarystyle{mcolindex}{%
9218 \csuse{@glscompstyle@index}%
9219 }%

```

Backward compatible mcolindexgroup style.

```

9220 \compatglossarystyle{mcolindexgroup}{%
9221 \csuse{@glscompstyle@index}%
9222 }%

```

Backward compatible mcolindexhypergroup style.

```

9223 \compatglossarystyle{mcolindexhypergroup}{%
9224 \csuse{@glscompstyle@index}%
9225 }%

```

Backward compatible mcoltree style.

```

9226 \compatglossarystyle{mcoltree}{%
9227 \csuse{@glscompstyle@tree}%
9228 }%

```

Backward compatible mcoltreegroup style.

```

9229 \compatglossarystyle{mcolindextreegroup}{%
9230 \csuse{@glscompstyle@tree}%
9231 }%

```

Backward compatible mcoltreehypergroup style.

```

9232 \compatglossarystyle{mcolindextreehypergroup}{%
9233 \csuse{@glscompstyle@tree}%
9234 }%

```

Backward compatible mcoltreenoname style.

```

9235 \compatglossarystyle{mcoltreenoname}{%
9236 \csuse{@glscompstyle@tree}%
9237 }%

```

Backward compatible mcoltreenonamegroup style.

```

9238 \compatglossarystyle{mcoltreenonamegroup}{%

```

```
9239 \csuse{@glscompstyle@tree}%
9240 }%
```

Backward compatible mcoltreenonamehypergroup style.

```
9241 \compatglossarystyle{mcoltreenonamehypergroup}{%
9242 \csuse{@glscompstyle@tree}%
9243 }%
```

Backward compatible mcolalmtree style.

```
9244 \compatglossarystyle{mcolalmtree}{%
9245 \csuse{@glscompstyle@almtree}%
9246 }%
```

Backward compatible mcolalmtreegroup style.

```
9247 \compatglossarystyle{mcolalmtreegroup}{%
9248 \csuse{@glscompstyle@almtree}%
9249 }%
```

Backward compatible mcolalmtreehypergroup style.

```
9250 \compatglossarystyle{mcolalmtreehypergroup}{%
9251 \csuse{@glscompstyle@almtree}%
9252 }%
```

Backward compatible superragged style.

```
9253 \compatglossarystyle{superragged}{%
9254 \renewcommand*{\glossaryentryfield}[5]{%
9255 \glsentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%
9256 \tabularnewline}%
9257 \renewcommand*{\glossarysubentryfield}[6]{%
9258 &
9259 \glssubentryitem{##2}%
9260 \glstarget{##2}{\strut}##4\glspostdescription\space ##6%
9261 \tabularnewline}%
9262 }%
```

Backward compatible superraggedborder style.

```
9263 \compatglossarystyle{superraggedborder}{%
9264 \csuse{@glscompstyle@superragged}%
9265 }%
```

Backward compatible superraggedheader style.

```
9266 \compatglossarystyle{superraggedheader}{%
9267 \csuse{@glscompstyle@superragged}%
9268 }%
```

Backward compatible superraggedheaderborder style.

```
9269 \compatglossarystyle{superraggedheaderborder}{%
9270 \csuse{@glscompstyle@superragged}%
9271 }%
```

Backward compatible superragged3col style.

```
9272 \compatglossarystyle{superragged3col}{%
9273 \renewcommand*{\glossaryentryfield}[5]{%
```

```

9274 \glentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\tabularnewline}%
9275 \renewcommand*\glossarysubentryfield}[6]{%
9276     &
9277     \glssubentryitem{##2}%
9278     \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%
9279 }%

```

Backward compatible superragged3colborder style.

```

9280 \compatglossarystyle{superragged3colborder}{%
9281 \csuse{@glscmpstyle@superragged3col}%
9282 }%

```

Backward compatible superragged3colheader style.

```

9283 \compatglossarystyle{superragged3colheader}{%
9284 \csuse{@glscmpstyle@superragged3col}%
9285 }%

```

Backward compatible superragged3colheaderborder style.

```

9286 \compatglossarystyle{superragged3colheaderborder}{%
9287 \csuse{@glscmpstyle@superragged3col}%
9288 }%

```

Backward compatible altsuperragged4col style.

```

9289 \compatglossarystyle{altsuperragged4col}{%
9290 \renewcommand*\glossaryentryfield}[5]{%
9291 \glentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\tabularnewline}%
9292 \renewcommand*\glossarysubentryfield}[6]{%
9293     &
9294     \glssubentryitem{##2}%
9295     \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%
9296 }%

```

Backward compatible altsuperragged4colheader style.

```

9297 \compatglossarystyle{altsuperragged4colheader}{%
9298 \csuse{@glscmpstyle@altsuperragged4col}%
9299 }%

```

Backward compatible altsuperragged4colborder style.

```

9300 \compatglossarystyle{altsuperragged4colborder}{%
9301 \csuse{@glscmpstyle@altsuperragged4col}%
9302 }%

```

Backward compatible altsuperragged4colheaderborder style.

```

9303 \compatglossarystyle{altsuperragged4colheaderborder}{%
9304 \csuse{@glscmpstyle@altsuperragged4col}%
9305 }%

```

Backward compatible super style.

```

9306 \compatglossarystyle{super}{%
9307 \renewcommand*\glossaryentryfield}[5]{%
9308 \glentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\}%
9309 \renewcommand*\glossarysubentryfield}[6]{%

```

```

9310      &
9311      \glssubentryitem{##2}%
9312      \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\}%
9313 }%

```

Backward compatible superborder style.

```

9314 \compatglossarystyle{superborder}{%
9315 \csuse{@glscmpstyle@super}%
9316 }%

```

Backward compatible superheader style.

```

9317 \compatglossarystyle{superheader}{%
9318 \csuse{@glscmpstyle@super}%
9319 }%

```

Backward compatible superheaderborder style.

```

9320 \compatglossarystyle{superheaderborder}{%
9321 \csuse{@glscmpstyle@super}%
9322 }%

```

Backward compatible super3col style.

```

9323 \compatglossarystyle{super3col}{%
9324 \renewcommand*{\glossaryentryfield}[5]{%
9325 \glstarget{##1}\glstarget{##1}{##2} & ##3 & ##5\\}%
9326 \renewcommand*{\glossarysubentryfield}[6]{%
9327 &
9328 \glssubentryitem{##2}%
9329 \glstarget{##2}{\strut}##4 & ##6\\}%
9330 }%

```

Backward compatible super3colborder style.

```

9331 \compatglossarystyle{super3colborder}{%
9332 \csuse{@glscmpstyle@super3col}%
9333 }%

```

Backward compatible super3colheader style.

```

9334 \compatglossarystyle{super3colheader}{%
9335 \csuse{@glscmpstyle@super3col}%
9336 }%

```

Backward compatible super3colheaderborder style.

```

9337 \compatglossarystyle{super3colheaderborder}{%
9338 \csuse{@glscmpstyle@super3col}%
9339 }%

```

Backward compatible super4col style.

```

9340 \compatglossarystyle{super4col}{%
9341 \renewcommand*{\glossaryentryfield}[5]{%
9342 \glstarget{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\\}%
9343 \renewcommand*{\glossarysubentryfield}[6]{%
9344 &
9345 \glssubentryitem{##2}%

```

```

9346 \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%
9347 }%

```

Backward compatible super4colheader style.

```

9348 \compatglossarystyle{super4colheader}{%
9349 \csuse{@glscompstyle@super4col}%
9350 }%

```

Backward compatible super4colborder style.

```

9351 \compatglossarystyle{super4colborder}{%
9352 \csuse{@glscompstyle@super4col}%
9353 }%

```

Backward compatible super4colheaderborder style.

```

9354 \compatglossarystyle{super4colheaderborder}{%
9355 \csuse{@glscompstyle@super4col}%
9356 }%

```

Backward compatible altsuper4col style.

```

9357 \compatglossarystyle{altsuper4col}{%
9358 \csuse{@glscompstyle@super4col}%
9359 }%

```

Backward compatible altsuper4colheader style.

```

9360 \compatglossarystyle{altsuper4colheader}{%
9361 \csuse{@glscompstyle@super4col}%
9362 }%

```

Backward compatible altsuper4colborder style.

```

9363 \compatglossarystyle{altsuper4colborder}{%
9364 \csuse{@glscompstyle@super4col}%
9365 }%

```

Backward compatible altsuper4colheaderborder style.

```

9366 \compatglossarystyle{altsuper4colheaderborder}{%
9367 \csuse{@glscompstyle@super4col}%
9368 }%

```

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.

```

9369 \NeedsTeXFormat{LaTeX2e}

```

Package version number now in line with main glossaries package number but will only be updated when glossaries-accsupp.sty is modified.

```

9370 \ProvidesPackage{glossaries-accsupp}[2014/03/06 v4.04 (NLCT)
9371 Experimental glossaries accessibility]

```

Pass all options to glossaries:

```

9372 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{glossaries}}

```

Process options:

```
9373 \ProcessOptions
```

ompatibleglossentry Override style compatibility macros:

```
9374 \def\compatibleglossentry#1#2{%
9375   \toks@{#2}%
9376   \protected@edef\@do@glossentry{%
9377     \noexpand\accsuppglossaryentryfield{#1}%
9378     {\noexpand\glsnamefont
9379      {\expandafter\expandonce\csname glo@\glsdetoklabel{#1}@name\endcsname}}}%
9380     {\expandafter\expandonce\csname glo@\glsdetoklabel{#1}@desc\endcsname}%
9381     {\expandafter\expandonce\csname glo@\glsdetoklabel{#1}@symbol\endcsname}%
9382     {\the\toks@}%
9383   }%
9384   \@do@glossentry
9385 }
```

atiblesubglossentry

```
9386 \def\compatiblesubglossentry#1#2#3{%
9387   \toks@{#3}%
9388   \protected@edef\@do@subglossentry{%
9389     \noexpand\accsuppglossarysubentryfield{\number#1}%
9390     {#2}%
9391     {\noexpand\glsnamefont
9392      {\expandafter\expandonce\csname glo@\glsdetoklabel{#2}@name\endcsname}}}%
9393     {\expandafter\expandonce\csname glo@\glsdetoklabel{#2}@desc\endcsname}%
9394     {\expandafter\expandonce\csname glo@\glsdetoklabel{#2}@symbol\endcsname}%
9395     {\the\toks@}%
9396   }%
9397   \@do@subglossentry
9398 }
```

Required packages:

```
9399 \RequirePackage{glossaries}
9400 \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:

```
9401 \define@key{glossentry}{access}{%
9402   \def\@glo@access{#1}%
9403 }
```

textaccess The replacement text corresponding to the text key:

```
9404 \define@key{glossentry}{textaccess}{%
9405   \def\@glo@textaccess{#1}%
9406 }
```

firstaccess The replacement text corresponding to the first key:

```
9407 \define@key{glossentry}{firstaccess}{%
9408   \def\@glo@firstaccess{#1}%
9409 }
```

pluralaccess The replacement text corresponding to the plural key:

```
9410 \define@key{glossentry}{pluralaccess}{%
9411   \def\@glo@pluralaccess{#1}%
9412 }
```

firstpluralaccess The replacement text corresponding to the firstplural key:

```
9413 \define@key{glossentry}{firstpluralaccess}{%
9414   \def\@glo@firstpluralaccess{#1}%
9415 }
```

symbolaccess The replacement text corresponding to the symbol key:

```
9416 \define@key{glossentry}{symbolaccess}{%
9417   \def\@glo@symbolaccess{#1}%
9418 }
```

symbolpluralaccess The replacement text corresponding to the symbolplural key:

```
9419 \define@key{glossentry}{symbolpluralaccess}{%
9420   \def\@glo@symbolpluralaccess{#1}%
9421 }
```

descriptionaccess The replacement text corresponding to the description key:

```
9422 \define@key{glossentry}{descriptionaccess}{%
9423   \def\@glo@descaccess{#1}%
9424 }
```

descriptionpluralaccess The replacement text corresponding to the descriptionplural key:

```
9425 \define@key{glossentry}{descriptionpluralaccess}{%
9426   \def\@glo@descpluralaccess{#1}%
9427 }
```

shortaccess The replacement text corresponding to the short key:

```
9428 \define@key{glossentry}{shortaccess}{%
9429   \def\@glo@shortaccess{#1}%
9430 }
```

shortpluralaccess The replacement text corresponding to the shortplural key:

```
9431 \define@key{glossentry}{shortpluralaccess}{%
9432   \def\@glo@shortpluralaccess{#1}%
9433 }
```

longaccess The replacement text corresponding to the long key:

```
9434 \define@key{glossentry}{longaccess}{%
9435   \def\@glo@longaccess{#1}%
9436 }
```

longpluralaccess The replacement text corresponding to the longplural key:

```
9437 \define@key{glossentry}{longpluralaccess}{%
9438   \def\@glo@longpluralaccess{#1}%
9439 }
```

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:

```
9440 \appto\@gls@keymap{,%
9441   {access}{access},%
9442   {textaccess}{textaccess},%
9443   {firstaccess}{firstaccess},%
9444   {pluralaccess}{pluralaccess},%
9445   {firstpluralaccess}{firstpluralaccess},%
9446   {symbolaccess}{symbolaccess},%
9447   {symbolpluralaccess}{symbolpluralaccess},%
9448   {descaccess}{descaccess},%
9449   {descpluralaccess}{descpluralaccess},%
9450   {shortaccess}{shortaccess},%
9451   {shortpluralaccess}{shortpluralaccess},%
9452   {longaccess}{longaccess},%
9453   {longpluralaccess}{longpluralaccess}%
9454 }
```

\@gls@noaccess Indicates that no replacement text has been provided.

```
9455 \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):

```
9456 \let\@gls@oldnewglossaryentryprehook\@newglossaryentryprehook
9457 \renewcommand*{\@newglossaryentryprehook}{%
9458   \@gls@oldnewglossaryentryprehook
9459   \def\@glo@access{\@glo@symbol}%
9460 }
```

Initialise the other keys:

```
9460 \def\@glo@textaccess{\@glo@access}%
9461 \def\@glo@firstaccess{\@glo@access}%
9462 \def\@glo@pluralaccess{\@glo@textaccess}%
9463 \def\@glo@firstpluralaccess{\@glo@pluralaccess}%
9464 \def\@glo@symbolaccess{\relax}%
9465 \def\@glo@symbolpluralaccess{\@glo@symbolaccess}%
9466 \def\@glo@descaccess{\relax}%
9467 \def\@glo@descpluralaccess{\@glo@descaccess}%
9468 }
```



```

9468 \def\@glo@shortaccess{\relax}%
9469 \def\@glo@shortpluralaccess{\@glo@shortaccess}%
9470 \def\@glo@longaccess{\relax}%
9471 \def\@glo@longpluralaccess{\@glo@longaccess}%
9472 }

```

Add to the end hook:

```

9473 \let\@gls@oldnewglossaryentryposthook\@newglossaryentryposthook
9474 \renewcommand*{\@newglossaryentryposthook}{%
9475   \@gls@oldnewglossaryentryposthook

```

Store the access information:

```

9476 \expandafter
9477   \protected@xdef\csname glo@\@glo@label @access\endcsname{%
9478     \@glo@access}%
9479 \expandafter
9480   \protected@xdef\csname glo@\@glo@label @textaccess\endcsname{%
9481     \@glo@textaccess}%
9482 \expandafter
9483   \protected@xdef\csname glo@\@glo@label @firstaccess\endcsname{%
9484     \@glo@firstaccess}%
9485 \expandafter
9486   \protected@xdef\csname glo@\@glo@label @pluralaccess\endcsname{%
9487     \@glo@pluralaccess}%
9488 \expandafter
9489   \protected@xdef\csname glo@\@glo@label @firstpluralaccess\endcsname{%
9490     \@glo@firstpluralaccess}%
9491 \expandafter
9492   \protected@xdef\csname glo@\@glo@label @symbolaccess\endcsname{%
9493     \@glo@symbolaccess}%
9494 \expandafter
9495   \protected@xdef\csname glo@\@glo@label @symbolpluralaccess\endcsname{%
9496     \@glo@symbolpluralaccess}%
9497 \expandafter
9498   \protected@xdef\csname glo@\@glo@label @descaccess\endcsname{%
9499     \@glo@descaccess}%
9500 \expandafter
9501   \protected@xdef\csname glo@\@glo@label @descpluralaccess\endcsname{%
9502     \@glo@descpluralaccess}%
9503 \expandafter
9504   \protected@xdef\csname glo@\@glo@label @shortaccess\endcsname{%
9505     \@glo@shortaccess}%
9506 \expandafter
9507   \protected@xdef\csname glo@\@glo@label @shortpluralaccess\endcsname{%
9508     \@glo@shortpluralaccess}%
9509 \expandafter
9510   \protected@xdef\csname glo@\@glo@label @longaccess\endcsname{%
9511     \@glo@longaccess}%
9512 \expandafter
9513   \protected@xdef\csname glo@\@glo@label @longpluralaccess\endcsname{%

```

```

9514      \@glo@longpluralaccess}%
9515 }

```

6.2 Accessing Replacement Text

`\glsentryaccess` Get the value of the access key for the entry with the given label:

```

9516 \newcommand*{\glsentryaccess}[1]{%
9517   \@gls@entry@field{#1}{access}%
9518 }

```

`\glsentrytextaccess` Get the value of the textaccess key for the entry with the given label:

```

9519 \newcommand*{\glsentrytextaccess}[1]{%
9520   \@gls@entry@field{#1}{textaccess}%
9521 }

```

`\glsentryfirstaccess` Get the value of the firstaccess key for the entry with the given label:

```

9522 \newcommand*{\glsentryfirstaccess}[1]{%
9523   \@gls@entry@field{#1}{firstaccess}%
9524 }

```

`\glsentrypluralaccess` Get the value of the pluralaccess key for the entry with the given label:

```

9525 \newcommand*{\glsentrypluralaccess}[1]{%
9526   \@gls@entry@field{#1}{pluralaccess}%
9527 }

```

`\glsentryfirstpluralaccess` Get the value of the firstpluralaccess key for the entry with the given label:

```

9528 \newcommand*{\glsentryfirstpluralaccess}[1]{%
9529   \csname glo@#1@firstpluralaccess\endcsname
9530 }

```

`\glsentrysymbolaccess` Get the value of the symbolaccess key for the entry with the given label:

```

9531 \newcommand*{\glsentrysymbolaccess}[1]{%
9532   \@gls@entry@field{#1}{symbolaccess}%
9533 }

```

`\glsentrysymbolpluralaccess` Get the value of the symbolpluralaccess key for the entry with the given label:

```

9534 \newcommand*{\glsentrysymbolpluralaccess}[1]{%
9535   \@gls@entry@field{#1}{symbolpluralaccess}%
9536 }

```

`\glsentrydescaccess` Get the value of the descriptionaccess key for the entry with the given label:

```

9537 \newcommand*{\glsentrydescaccess}[1]{%
9538   \@gls@entry@field{#1}{descaccess}%
9539 }

```

`\glsentrydescpluralaccess` Get the value of the descriptionpluralaccess key for the entry with the given label:

```

9540 \newcommand*{\glsentrydescpluralaccess}[1]{%
9541   \@gls@entry@field{#1}{descaccess}%
9542 }

```

`\glsentryshortaccess` Get the value of the shortaccess key for the entry with the given label:

```

9543 \newcommand*{\glsentryshortaccess}[1]{%
9544   \@gls@entry@field{#1}{shortaccess}%
9545 }

```

`\glsentryshortpluralaccess` Get the value of the shortpluralaccess key for the entry with the given label:

```

9546 \newcommand*{\glsentryshortpluralaccess}[1]{%
9547   \@gls@entry@field{#1}{shortpluralaccess}%
9548 }

```

`\glsentrylongaccess` Get the value of the longaccess key for the entry with the given label:

```

9549 \newcommand*{\glsentrylongaccess}[1]{%
9550   \@gls@entry@field{#1}{longaccess}%
9551 }

```

`\glsentrylongpluralaccess` Get the value of the longpluralaccess key for the entry with the given label:

```

9552 \newcommand*{\glsentrylongpluralaccess}[1]{%
9553   \@gls@entry@field{#1}{longpluralaccess}%
9554 }

```

`\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.)

```

9555 \newcommand*{\glsaccsupp}[2]{%
9556   \BeginAccSupp{ActualText=#1}#2\EndAccSupp{}%
9557 }

```

`\xglsaccsupp` Fully expands replacement text before calling `\glsaccsupp`

```

9558 \newcommand*{\xglsaccsupp}[2]{%
9559   \protected@edef\@gls@replacementtext{#1}%
9560   \expandafter\glsaccsupp\expandafter{\@gls@replacementtext}{#2}%
9561 }

```

`\@gls@access@display`

```

9562 \newcommand*{\@gls@access@display}[2]{%
9563   \protected@edef\@glo@access{#2}%
9564   \ifx\@glo@access\@gls@noaccess
9565     #1%
9566   \else
9567     \xglsaccsupp{\@glo@access}{#1}%
9568   \fi
9569 }

```

`l$nameaccessdisplay` Displays the first argument with the accessibility text for the entry with the label given by the second argument (if set).

```
9570 \DeclareRobustCommand*{\gl$nameaccessdisplay}[2]{%
9571   \@gls@access@display{#1}{\gl$entryaccess{#2}}}%
9572 }
```

`l$textaccessdisplay` As above but for the `textaccess` replacement text.

```
9573 \DeclareRobustCommand*{\gl$textaccessdisplay}[2]{%
9574   \@gls@access@display{#1}{\gl$entrytextaccess{#2}}}%
9575 }
```

`pluralaccessdisplay` As above but for the `pluralaccess` replacement text.

```
9576 \DeclareRobustCommand*{\gl$pluralaccessdisplay}[2]{%
9577   \@gls@access@display{#1}{\gl$entrypluralaccess{#2}}}%
9578 }
```

`sfirstaccessdisplay` As above but for the `firstaccess` replacement text.

```
9579 \DeclareRobustCommand*{\gl$sfirstaccessdisplay}[2]{%
9580   \@gls@access@display{#1}{\gl$entryfirstaccess{#2}}}%
9581 }
```

`pluralaccessdisplay` As above but for the `firstpluralaccess` replacement text.

```
9582 \DeclareRobustCommand*{\gl$sfirstpluralaccessdisplay}[2]{%
9583   \@gls@access@display{#1}{\gl$entryfirstpluralaccess{#2}}}%
9584 }
```

`symbolaccessdisplay` As above but for the `symbolaccess` replacement text.

```
9585 \DeclareRobustCommand*{\gl$symbolaccessdisplay}[2]{%
9586   \@gls@access@display{#1}{\gl$entrysymbolaccess{#2}}}%
9587 }
```

`pluralaccessdisplay` As above but for the `symbolpluralaccess` replacement text.

```
9588 \DeclareRobustCommand*{\gl$symbolpluralaccessdisplay}[2]{%
9589   \@gls@access@display{#1}{\gl$entrysymbolpluralaccess{#2}}}%
9590 }
```

`l$ptionaccessdisplay` As above but for the `descriptionaccess` replacement text.

```
9591 \DeclareRobustCommand*{\gl$descriptionaccessdisplay}[2]{%
9592   \@gls@access@display{#1}{\gl$entrydescaccess{#2}}}%
9593 }
```

`pluralaccessdisplay` As above but for the `descriptionpluralaccess` replacement text.

```
9594 \DeclareRobustCommand*{\gl$descriptionpluralaccessdisplay}[2]{%
9595   \@gls@access@display{#1}{\gl$entrydescpluralaccess{#2}}}%
9596 }
```

`sshortaccessdisplay` As above but for the `shortaccess` replacement text.

```

9597 \DeclareRobustCommand*\glsshortaccessdisplay}[2]{%
9598   \@gls@access@display{#1}{\glsentryshortaccess{#2}}%
9599 }

```

`pluralaccessdisplay` As above but for the `shortpluralaccess` replacement text.

```

9600 \DeclareRobustCommand*\glsshortpluralaccessdisplay}[2]{%
9601   \@gls@access@display{#1}{\glsentryshortpluralaccess{#2}}%
9602 }

```

`longaccessdisplay` As above but for the `longaccess` replacement text.

```

9603 \DeclareRobustCommand*\glslongaccessdisplay}[2]{%
9604   \@gls@access@display{#1}{\glsentrylongaccess{#2}}%
9605 }

```

`longpluralaccessdisplay` As above but for the `longpluralaccess` replacement text.

```

9606 \DeclareRobustCommand*\glslongpluralaccessdisplay}[2]{%
9607   \@gls@access@display{#1}{\glsentrylongpluralaccess{#2}}%
9608 }

```

`\glsaccessdisplay` Gets the replacement text corresponding to the named key given by the first argument and calls the appropriate command defined above.

```

9609 \DeclareRobustCommand*\glsaccessdisplay}[3]{%
9610   \@ifundefined{gls#1accessdisplay}%
9611   {%
9612     \PackageError{glossaries-accsupp}{No accessibility support
9613       for key ‘#1’}{}%
9614   }%
9615   {%
9616     \csname gls#1accessdisplay\endcsname{#2}{#3}%
9617   }%
9618 }

```

`ls@default@entryfmt` Redefine the default entry format to use accessibility information

```

9619 \renewcommand*\@@gls@default@entryfmt}[2]{%
9620   \ifdefempty\glscustomtext
9621   {%
9622     \glsifplural
9623     {%

```

Plural form

```

9624       \glscapscase
9625       {%

```

Don't adjust case

```

9626       \ifglsused\glslabel
9627       {%

```

Subsequent use

```

9628      #2{\glspluralaccessdisplay
9629          {\glsentryplural{\glslabel}}{\glslabel}}%
9630          {\glsdescriptionpluralaccessdisplay
9631              {\glsentrydescplural{\glslabel}}{\glslabel}}%
9632          {\glssymbolpluralaccessdisplay
9633              {\glsentrysymbolplural{\glslabel}}{\glslabel}}
9634          {\glsinsert}}%
9635      }%
9636      {%

```

First use

```

9637      #1{\glsfirstpluralaccessdisplay
9638          {\glsentryfirstplural{\glslabel}}{\glslabel}}%
9639          {\glsdescriptionpluralaccessdisplay
9640              {\glsentrydescplural{\glslabel}}{\glslabel}}%
9641          {\glssymbolpluralaccessdisplay
9642              {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
9643          {\glsinsert}}%
9644      }%
9645      }%
9646      {%

```

Make first letter upper case

```

9647      \ifglsused\glslabel
9648      {%

```

Subsequent use.

```

9649      #2{\glspluralaccessdisplay
9650          {\Glsentryplural{\glslabel}}{\glslabel}}%
9651          {\glsdescriptionpluralaccessdisplay
9652              {\glsentrydescplural{\glslabel}}{\glslabel}}%
9653          {\glssymbolpluralaccessdisplay
9654              {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
9655          {\glsinsert}}%
9656      }%
9657      {%

```

First use

```

9658      #1{\glsfirstpluralaccessdisplay
9659          {\Glsentryfirstplural{\glslabel}}{\glslabel}}%
9660          {\glsdescriptionpluralaccessdisplay
9661              {\glsentrydescplural{\glslabel}}{\glslabel}}%
9662          {\glssymbolpluralaccessdisplay
9663              {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
9664          {\glsinsert}}%
9665      }%
9666      }%
9667      {%

```

Make all upper case

```

9668      \ifglused\glslabel
9669      {%

```

Subsequent use

```

9670      \MakeUppercase{%
9671      #2{\glspluralaccessdisplay
9672      {\glentryplural{\glslabel}}{\glslabel}}%
9673      {\gldescriptionpluralaccessdisplay
9674      {\glentrydescplural{\glslabel}}{\glslabel}}%
9675      {\glssymbolpluralaccessdisplay
9676      {\glentrysymbolplural{\glslabel}}{\glslabel}}%
9677      {\glinsert}}%
9678      }%
9679      {%

```

First use

```

9680      \MakeUppercase{%
9681      #1{\glfirstpluralaccessdisplay
9682      {\glentryfirstplural{\glslabel}}{\glslabel}}%
9683      {\gldescriptionpluralaccessdisplay
9684      {\glentrydescplural{\glslabel}}{\glslabel}}%
9685      {\glssymbolpluralaccessdisplay
9686      {\glentrysymbolplural{\glslabel}}{\glslabel}}%
9687      {\glinsert}}%
9688      }%
9689      }%
9690      }%
9691      {%

```

Singular form

```

9692      \glscapscase
9693      {%

```

Don't adjust case

```

9694      \ifglused\glslabel
9695      {%

```

Subsequent use

```

9696      #2{\glstextaccessdisplay
9697      {\glentrytext{\glslabel}}{\glslabel}}%
9698      {\gldescriptionaccessdisplay
9699      {\glentrydesc{\glslabel}}{\glslabel}}%
9700      {\glssymbolaccessdisplay
9701      {\glentrysymbol{\glslabel}}{\glslabel}}%
9702      {\glinsert}}%
9703      }%
9704      {%

```

First use

```

9705      #1{\glfirstaccessdisplay
9706      {\glentryfirst{\glslabel}}{\glslabel}}%
9707      {\gldescriptionaccessdisplay

```

```

9708         {\glentrydesc{\glslabel}}{\glslabel}}%
9709         {\glssymbolaccessdisplay
9710         {\glentrysymbol{\glslabel}}{\glslabel}}}%
9711         {\glsinsert}%
9712     }%
9713 }%
9714 {%

```

Make first letter upper case

```

9715     \ifglsused\glslabel
9716     {%

```

Subsequent use

```

9717     #2{\glstextaccessdisplay
9718         {\Glsentrytext{\glslabel}}{\glslabel}}%
9719         {\glsdescriptionaccessdisplay
9720         {\glentrydesc{\glslabel}}{\glslabel}}%
9721         {\glssymbolaccessdisplay
9722         {\glentrysymbol{\glslabel}}{\glslabel}}}%
9723         {\glsinsert}%
9724     }%
9725     {%

```

First use

```

9726     #1{\glsfirstaccessdisplay
9727         {\Glsentryfirst{\glslabel}}{\glslabel}}%
9728         {\glsdescriptionaccessdisplay
9729         {\glentrydesc{\glslabel}}{\glslabel}}%
9730         {\glssymbolaccessdisplay
9731         {\glentrysymbol{\glslabel}}{\glslabel}}}%
9732         {\glsinsert}%
9733     }%
9734 }%
9735 {%

```

Make all upper case

```

9736     \ifglsused\glslabel
9737     {%

```

Subsequent use

```

9738     \MakeUppercase{%
9739     #2{\glstextaccessdisplay
9740         {\glentrytext{\glslabel}}{\glslabel}}%
9741         {\glsdescriptionaccessdisplay
9742         {\glentrydesc{\glslabel}}{\glslabel}}%
9743         {\glssymbolaccessdisplay
9744         {\glentrysymbol{\glslabel}}{\glslabel}}}%
9745         {\glsinsert}}%
9746     }%
9747     {%

```

First use


```

9748         \MakeUppercase{%
9749         #1{\glsfirstaccessdisplay
9750             {\glsentryfirst{\glslabel}}{\glslabel}}%
9751             {\glsdescriptionaccessdisplay
9752             {\glsentrydesc{\glslabel}}{\glslabel}}%
9753             {\glssymbolaccessdisplay
9754             {\glsentrysymbol{\glslabel}}{\glslabel}}%
9755             {\glsinsert}}%
9756     }%
9757 }%
9758 }%
9759 }%
9760 {%

```

Custom text provided in \glsdisp

```

9761     \ifglsused{\glslabel}%
9762     {%

```

Subsequent use

```

9763     #2{\glscustomtext}%
9764     {\glsdescriptionaccessdisplay
9765     {\glsentrydesc{\glslabel}}{\glslabel}}%
9766     {\glssymbolaccessdisplay
9767     {\glsentrysymbol{\glslabel}}{\glslabel}}%
9768     {\glsinsert}}%
9769 }%
9770 {%

```

First use

```

9771     #1{\glscustomtext}%
9772     {\glsdescriptionaccessdisplay
9773     {\glsentrydesc{\glslabel}}{\glslabel}}%
9774     {\glssymbolaccessdisplay
9775     {\glsentrysymbol{\glslabel}}{\glslabel}}%
9776     {\glsinsert}}%
9777 }%
9778 }%
9779 }

```

\glsгенentryfmt Redefine to use accessibility information.

```

9780 \renewcommand*{\glsгенentryfmt}{%
9781     \ifdefempty\glscustomtext
9782     {%
9783         \glsifplural
9784         {%

```

Plural form

```

9785         \glscapscase
9786         {%

```

Don't adjust case

```

9787      \ifglsused\glslabel
9788      {%

```

Subsequent use

```

9789      \glspluralaccessdisplay
9790      {\glsentryplural{\glslabel}}{\glslabel}%
9791      \glsinsert
9792      }%
9793      {%

```

First use

```

9794      \glsfirstpluralaccessdisplay
9795      {\glsentryfirstplural{\glslabel}}{\glslabel}%
9796      \glsinsert
9797      }%
9798      }%
9799      {%

```

Make first letter upper case

```

9800      \ifglsused\glslabel
9801      {%

```

Subsequent use.

```

9802      \glspluralaccessdisplay
9803      {\Glsentryplural{\glslabel}}{\glslabel}%
9804      \glsinsert
9805      }%
9806      {%

```

First use

```

9807      \glsfirstpluralaccessdisplay
9808      {\Glsentryfirstplural{\glslabel}}{\glslabel}%
9809      \glsinsert
9810      }%
9811      }%
9812      {%

```

Make all upper case

```

9813      \ifglsused\glslabel
9814      {%

```

Subsequent use

```

9815      \glspluralaccessdisplay
9816      {\mfirstucMakeUppercase{\glsentryplural{\glslabel}}}%
9817      {\glslabel}%
9818      \mfirstucMakeUppercase{\glsinsert}%
9819      }%
9820      {%

```

First use

```

9821      \glsfirstpluralaccessdisplay
9822      {\mfirstucMakeUppercase{\glsentryfirstplural{\glslabel}}}%

```

```

9823         {\glslabel}%
9824     \mfirstucMakeUppercase{\glsinsert}%
9825 }%
9826 }%
9827 }%
9828 {%

```

Singular form

```

9829     \glscapscase
9830 {%

```

Don't adjust case

```

9831     \ifglsused\glslabel
9832 {%

```

Subsequent use

```

9833     \glstextaccessdisplay{\glsentrytext{\glslabel}}{\glslabel}%
9834     \glsinsert
9835 }%
9836 {%

```

First use

```

9837     \glsfirstaccessdisplay{\glsentryfirst{\glslabel}}{\glslabel}%
9838     \glsinsert
9839 }%
9840 }%
9841 {%

```

Make first letter upper case

```

9842     \ifglsused\glslabel
9843 {%

```

Subsequent use

```

9844     \glstextaccessdisplay{\Glsentrytext{\glslabel}}{\glslabel}%
9845     \glsinsert
9846 }%
9847 {%

```

First use

```

9848     \glsfirstaccessdisplay{\Glsentryfirst{\glslabel}}{\glslabel}%
9849     \glsinsert
9850 }%
9851 }%
9852 {%

```

Make all upper case

```

9853     \ifglsused\glslabel
9854 {%

```

Subsequent use

```

9855     \glstextaccessdisplay
9856     {\mfirstucMakeUppercase{\glsentrytext{\glslabel}}}{\glslabel}%
9857     \mfirstucMakeUppercase{\glsinsert}%

```

```

9858         }%
9859         {%

First use
9860         \glsfirstaccessdisplay
9861         {\mfirstucMakeUppercase{\glstryfirst{\glslabel}}}{\glslabel}%
9862         \mfirstucMakeUppercase{\glsinsert}%
9863         }%
9864     }%
9865 }%
9866 }%
9867 {%

```

Custom text provided in `\glsdisp`. (The insert should be empty at this point.)
The accessibility information, if required, will have to be explicitly included in the custom text.

```

9868     \glscustomtext\glsinsert
9869 }%
9870 }

```

`\glsacnfont` Redefine to include accessibility information.

```

9871 \renewcommand*{\glsacnfont}{%
9872     \ifempty\glscustomtext
9873     {%
9874         \ifglused\glslabel
9875         {%

```

Subsequent use:

```

9876     \glsifplural
9877     {%

```

Subsequent plural form:

```

9878     \glscapscase
9879     {%

```

Subsequent plural form, don't adjust case:

```

9880     \acronymfont
9881     {\glsshortpluralaccessdisplay
9882     {\glstryshortpl{\glslabel}}{\glslabel}}%
9883     \glsinsert
9884     }%
9885     {%

```

Subsequent plural form, make first letter upper case:

```

9886     \acronymfont
9887     {\glsshortpluralaccessdisplay
9888     {\Glsentryshortpl{\glslabel}}{\glslabel}}%
9889     \glsinsert
9890     }%
9891     {%

```

Subsequent plural form, all caps:

```

9892      \mfirstucMakeUppercase
9893      {\acronymfont
9894      {\glsshortpluralaccessdisplay
9895      {\glentryshortpl{\glslabel}}{\glslabel}}%
9896      \glsinsert}%
9897      }%
9898      }%
9899      {%

```

Subsequent singular form

```

9900      \glscapscase
9901      {%

```

Subsequent singular form, don't adjust case:

```

9902      \acronymfont
9903      {\glsshortaccessdisplay{\glentryshort{\glslabel}}{\glslabel}}%
9904      \glsinsert
9905      }%
9906      {%

```

Subsequent singular form, make first letter upper case:

```

9907      \acronymfont
9908      {\glsshortaccessdisplay{\Glentryshort{\glslabel}}{\glslabel}}%
9909      \glsinsert
9910      }%
9911      {%

```

Subsequent singular form, all caps:

```

9912      \mfirstucMakeUppercase
9913      {\acronymfont{%
9914      \glsshortaccessdisplay{\glentryshort{\glslabel}}{\glslabel}}%
9915      \glsinsert}%
9916      }%
9917      }%
9918      }%
9919      {%

```

First use:

```

9920      \glsifplural
9921      {%

```

First use plural form:

```

9922      \glscapscase
9923      {%

```

First use plural form, don't adjust case:

```

9924      \genplacrfullformat{\glslabel}{\glsinsert}%
9925      }%
9926      {%

```

First use plural form, make first letter upper case:

```
9927      \Genplacrfullformat{\glslabel}{\glsinsert}%
9928      }%
9929      {%
```

First use plural form, all caps:

```
9930      \mfirstucMakeUppercase
9931      {\genplacrfullformat{\glslabel}{\glsinsert}}%
9932      }%
9933      }%
9934      {%
```

First use singular form

```
9935      \glscapscase
9936      {%
```

First use singular form, don't adjust case:

```
9937      \genacrfullformat{\glslabel}{\glsinsert}%
9938      }%
9939      {%
```

First use singular form, make first letter upper case:

```
9940      \Genacrfullformat{\glslabel}{\glsinsert}%
9941      }%
9942      {%
```

First use singular form, all caps:

```
9943      \mfirstucMakeUppercase
9944      {\genacrfullformat{\glslabel}{\glsinsert}}%
9945      }%
9946      }%
9947      }%
9948      }%
9949      {%
```

User supplied text. (The insert should be empty at this point.) The accessibility information, if required, will have to be explicitly included in the custom text.

```
9950      \glscustomtext
9951      }%
9952 }
```

`\genacrfullformat` Redefine to include accessibility information.

```
9953 \renewcommand*{\genacrfullformat}[2]{%
9954   \glslongaccessdisplay{\glsentrylong{#1}}{#1}#2\space
9955   (\glsshortaccessdisplay{\protect\firstacronymfont{\glsentryshort{#1}}}{#1})%
9956 }
```

`\Genacrfullformat` Redefine to include accessibility information.

```
9957 \renewcommand*{\Genacrfullformat}[2]{%
9958   \glslongaccessdisplay{\Glsentrylong{#1}}{#1}#2\space
9959   (\glsshortaccessdisplay{\protect\firstacronymfont{\Glsentryshort{#1}}}{#1})%
9960 }
```

`\genplacrfullformat` Redefine to include accessibility information.

```
9961 \renewcommand*{\genplacrfullformat}[2]{%
9962   \glslongpluralaccessdisplay{\glsentrylongpl{#1}}{#1}#2\space
9963   (\glsshortpluralaccessdisplay
9964     {\protect\firstacronymfont{\glsentryshortpl{#1}}{#1}})%
9965 }
```

`\Genplacrfullformat` Redefine to include accessibility information.

```
9966 \renewcommand*{\Genplacrfullformat}[2]{%
9967   \glslongpluralaccessdisplay{\Glsentrylongpl{#1}}{#1}#2\space
9968   (\glsshortpluralaccessdisplay
9969     {\protect\firstacronymfont{\glsentryshortpl{#1}}{#1}})%
9970 }
```

`\@acrshort`

```
9971 \def\@acrshort#1#2[#3]{%
9972   \glsdoifexists{#2}%
9973   {%
9974     \edef\@glo@type{\glsentrytype{#2}}%

9975     \let\glsifplural\@secondoftwo
9976     \let\glscapscase\@firstofthree
9977     \let\glsinsert\@empty
9978     \def\glscustomtext{%
9979       \acronymfont{\glsshortaccessdisplay{\glsentryshort{#2}}{#2}}#3%
9980     }%
```

Call `\@gls@link`

```
9981   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
9982   }%
9983 }
```

`\@Acrshort`

```
9984 \def\@Acrshort#1#2[#3]{%
9985   \glsdoifexists{#2}%
9986   {%
9987     \edef\@glo@type{\glsentrytype{#2}}%

9988     \let\glsifplural\@secondoftwo
9989     \let\glscapscase\@secondofthree
9990     \let\glsinsert\@empty
9991     \def\glscustomtext{%
9992       \acronymfont{\glsshortaccessdisplay{\Glsentryshort{#2}}{#2}}#3%
9993     }%
```

Call `\@gls@link`

```
9994   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
9995   }%
9996 }
```

\@ACRshort

```

9997 \def\@ACRshort#1#2[#3]{%
9998   \glsdoifexists{#2}%
9999   {%
10000     \edef\@glo@type{\glentrytype{#2}}%

10001     \let\glsifplural\@secondoftwo
10002     \let\glscapscase\@thirdofthree
10003     \let\glsinsert\@empty
10004     \def\glscustomtext{%
10005       \acronymfont{\glsshortaccessdisplay
10006         {\MakeUppercase{\glentryshort{#2}}}{#2}}#3%
10007     }%

    Call \@gls@link
10008     \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
10009   }%
10010 }

```

\@acrlong

```

10011 \def\@acrlong#1#2[#3]{%
10012   \glsdoifexists{#2}%
10013   {%
10014     \edef\@glo@type{\glentrytype{#2}}%

10015     \let\glsifplural\@secondoftwo
10016     \let\glscapscase\@firstofthree
10017     \let\glsinsert\@empty
10018     \def\glscustomtext{%
10019       \acronymfont{\glslongaccessdisplay{\glentrylong{#2}}{#2}}#3%
10020     }%

    Call \@gls@link
10021     \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
10022   }%
10023 }

```

\@Acrlong

```

10024 \def\@Acrlong#1#2[#3]{%
10025   \glsdoifexists{#2}%
10026   {%
10027     \edef\@glo@type{\glentrytype{#2}}%

10028     \let\glsifplural\@secondoftwo
10029     \let\glscapscase\@firstofthree
10030     \let\glsinsert\@empty
10031     \def\glscustomtext{%
10032       \acronymfont{\glslongaccessdisplay{\Glsentrylong{#2}}{#2}}#3%
10033     }%

```



```

Call \@gls@link
10034   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
10035   }%
10036 }

```

\@ACRlong

```

10037 \def\@ACRlong#1#2[#3]{%
10038   \glsdoifexists{#2}%
10039   {%
10040     \edef\@glo@type{\glentrytype{#2}}%

10041     \let\glsifplural\@secondoftwo
10042     \let\glscapscase\@firstofthree
10043     \let\glsinsert\@empty
10044     \def\glscustomtext{%
10045       \acronymfont{\glslongaccessdisplay{%
10046         \MakeUppercase{\glentrylong{#2}}}{#2}#3}%
10047     }%

```

```

Call \@gls@link
10048   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
10049   }%
10050 }

```

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.

```

10051 \renewcommand*{\glossentryname}[1]{%
10052   \glsdoifexists{#1}%
10053   {%
10054     \glsnamefont{\glsnameaccessdisplay{\glentryname{#1}}{#1}}%
10055   }%
10056 }

10057 \renewcommand*{\glossentryname}[1]{%
10058   \glsdoifexists{#1}%
10059   {%
10060     \glsnamefont{\glsnameaccessdisplay{\Glsentryname{#1}}{#1}}%
10061   }%
10062 }

10063 \renewcommand*{\glossentrydesc}[1]{%
10064   \glsdoifexists{#1}%

```

```

10065  {%
10066    \glsdescriptionaccessdisplay{\glsentrydesc{#1}}{#1}%
10067  }%
10068 }

10069 \renewcommand*{\Glossentrydesc}[1]{%
10070   \glsdoifexists{#1}%
10071   {%
10072     \glsdescriptionaccessdisplay{\Glsentrydesc{#1}}{#1}%
10073   }%
10074 }

10075 \renewcommand*{\glossentrysymbol}[1]{%
10076   \glsdoifexists{#1}%
10077   {%
10078     \glssymbolaccessdisplay{\glsentrysymbol{#1}}{#1}%
10079   }%
10080 }

10081 \renewcommand*{\Glossentrysymbol}[1]{%
10082   \glsdoifexists{#1}%
10083   {%
10084     \glssymbolaccessdisplay{\Glsentrysymbol{#1}}{#1}%
10085   }%
10086 }

```

pglossaryentryfield

```

10087 \newcommand*{\accsuppglossaryentryfield}[5]{%
10088   \glossaryentryfield{#1}%
10089   {\glsnameaccessdisplay{#2}{#1}}%
10090   {\glsdescriptionaccessdisplay{#3}{#1}}%
10091   {\glssymbolaccessdisplay{#4}{#1}}{#5}%
10092 }

```

glossarysubentryfield

```

10093 \newcommand*{\accsuppglossarysubentryfield}[6]{%
10094   \glossarysubentryfield{#1}{#2}%
10095   {\glsnameaccessdisplay{#3}{#2}}%
10096   {\glsdescriptionaccessdisplay{#4}{#2}}%
10097   {\glssymbolaccessdisplay{#5}{#2}}{#6}%
10098 }

```

6.4 Acronyms

Redefine acronym styles provided by glossaries:

`long-short` $\langle long \rangle$ ($\langle short \rangle$) acronym style.

```

10099 \renewacronymstyle{long-short}%
10100 {%

```

Check for long form in case this is a mixed glossary.

```

10101 \ifglshaslong{\glslabel}{\glsacronymfont}{\glsacronymentryfont}%
10102 }%
10103 {%
10104 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
10105 \renewcommand*{\genacrfullformat}[2]{%
10106 \glslongaccessdisplay{\glsentrylong{##1}}{##1}##2\space
10107 (\glsshortaccessdisplay
10108 {\protect\firstacronymfont{\glsentryshort{##1}}}{##1})%
10109 }%
10110 \renewcommand*{\Genacrfullformat}[2]{%
10111 \glslongaccessdisplay{\Glsentrylong{##1}}{##1}##2\space
10112 (\glsshortaccessdisplay
10113 {\protect\firstacronymfont{\Glsentryshort{##1}}}{##1})%
10114 }%
10115 \renewcommand*{\genplacrfullformat}[2]{%
10116 \glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1}##2\space
10117 (\glsshortpluralaccessdisplay
10118 {\protect\firstacronymfont{\glsentryshortpl{##1}}}{##1})%
10119 }%
10120 \renewcommand*{\Genplacrfullformat}[2]{%
10121 \glslongpluralaccessdisplay{\Glsentrylongpl{##1}}{##1}##2\space
10122 (\glsshortpluralaccessdisplay
10123 {\protect\firstacronymfont{\Glsentryshortpl{##1}}}{##1})%
10124 }%
10125 \renewcommand*{\acronymentry}[1]{%
10126 \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}}
10127 \renewcommand*{\acronymentry}[2]{##1}%
10128 \renewcommand*{\acronymfont}[1]{##1}%
10129 \renewcommand*{\firstacronymfont}[1]{\acronymfont{##1}}%
10130 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
10131 }

```

short-long <short> (<long>) acronym style.

```

10132 \renewacronymstyle{short-long}%
10133 {%

```

Check for long form in case this is a mixed glossary.

```

10134 \ifglshaslong{\glslabel}{\glsacronymfont}{\glsacronymentryfont}%
10135 }%
10136 {%
10137 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
10138 \renewcommand*{\genacrfullformat}[2]{%
10139 \glsshortaccessdisplay
10140 {\protect\firstacronymfont{\glsentryshort{##1}}}{##1}##2\space
10141 (\glslongaccessdisplay{\glsentrylong{##1}}{##1})%
10142 }%
10143 \renewcommand*{\Genacrfullformat}[2]{%
10144 \glsshortaccessdisplay

```

```

10145      {\protect\firstacronymfont{\Glsentryshort{##1}}}{##1}##2\space
10146      (\glslongaccessdisplay{\glsentrylong{##1}}{##1})%
10147  }%
10148  \renewcommand*{\genplacrfullformat}[2]{%
10149    \glsshortpluralaccessdisplay
10150    {\protect\firstacronymfont{\glsentryshortpl{##1}}}{##1}##2\space
10151    (\glslongpluralaccessdisplay
10152    {\glsentrylongpl{##1}}{##1})%
10153  }%
10154  \renewcommand*{\Genplacrfullformat}[2]{%
10155    \glsshortpluralaccessdisplay
10156    {\protect\firstacronymfont{\Glsentryshortpl{##1}}}{##1}##2\space
10157    (\glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1})%
10158  }%
10159  \renewcommand*{\acronymentry}[1]{%
10160    \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}%
10161  \renewcommand*{\acronymsort}[2]{##1}%
10162  \renewcommand*{\acronymfont}[1]{##1}%
10163  \renewcommand*{\firstacronymfont}[1]{\acronymfont{##1}}%
10164  \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
10165  }

```

long-short-desc *<long>* (*<short>*) acronym style that has an accompanying description (which the user needs to supply).

```

10166 \renewacronymstyle{long-short-desc}%
10167 {%
10168   \GlsUseAcrEntryDispStyle{long-short}%
10169 }%
10170 {%
10171   \GlsUseAcrStyleDefs{long-short}%
10172   \renewcommand*{\GenericAcronymFields}{}%
10173   \renewcommand*{\acronymsort}[2]{##2}%
10174   \renewcommand*{\acronymentry}[1]{%
10175     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10176     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10177 }

```

long-sc-short-desc *<long>* (*\textsc{<short>}*) acronym style that has an accompanying description (which the user needs to supply).

```

10178 \renewacronymstyle{long-sc-short-desc}%
10179 {%
10180   \GlsUseAcrEntryDispStyle{long-sc-short}%
10181 }%
10182 {%
10183   \GlsUseAcrStyleDefs{long-sc-short}%
10184   \renewcommand*{\GenericAcronymFields}{}%
10185   \renewcommand*{\acronymsort}[2]{##2}%
10186   \renewcommand*{\acronymentry}[1]{%
10187     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space

```

```

10188      (\glsshortaccessdisplay{\acronymfont{\glentryshort{##1}}}{##1}})%
10189 }

```

long-sm-short-desc *⟨long⟩* (\textsmaller{*⟨short⟩*}) acronym style that has an accompanying description (which the user needs to supply).

```

10190 \renewacronymstyle{long-sm-short-desc}%
10191 {%
10192   \GlsUseAcrEntryDisplayStyle{long-sm-short}%
10193 }%
10194 {%
10195   \GlsUseAcrStyleDefs{long-sm-short}%
10196   \renewcommand*{\GenericAcronymFields}{}%
10197   \renewcommand*{\acronymsort}[2]{##2}%
10198   \renewcommand*{\acronymentry}[1]{%
10199     \glslongaccessdisplay{\glentrylong{##1}}{##1}\space
10200     (\glsshortaccessdisplay{\acronymfont{\glentryshort{##1}}}{##1}})%
10201 }

```

short-long-desc *⟨short⟩* ({*⟨long⟩*}) acronym style that has an accompanying description (which the user needs to supply).

```

10202 \renewacronymstyle{short-long-desc}%
10203 {%
10204   \GlsUseAcrEntryDisplayStyle{short-long}%
10205 }%
10206 {%
10207   \GlsUseAcrStyleDefs{short-long}%
10208   \renewcommand*{\GenericAcronymFields}{}%
10209   \renewcommand*{\acronymsort}[2]{##2}%
10210   \renewcommand*{\acronymentry}[1]{%
10211     \glslongaccessdisplay{\glentrylong{##1}}{##1}\space
10212     (\glsshortaccessdisplay{\acronymfont{\glentryshort{##1}}}{##1}})%
10213 }

```

sc-short-long-desc *⟨long⟩* (\textsc{*⟨short⟩*}) acronym style that has an accompanying description (which the user needs to supply).

```

10214 \renewacronymstyle{sc-short-long-desc}%
10215 {%
10216   \GlsUseAcrEntryDisplayStyle{sc-short-long}%
10217 }%
10218 {%
10219   \GlsUseAcrStyleDefs{sc-short-long}%
10220   \renewcommand*{\GenericAcronymFields}{}%
10221   \renewcommand*{\acronymsort}[2]{##2}%
10222   \renewcommand*{\acronymentry}[1]{%
10223     \glslongaccessdisplay{\glentrylong{##1}}{##1}\space
10224     (\glsshortaccessdisplay{\acronymfont{\glentryshort{##1}}}{##1}})%
10225 }

```

sm-short-long-desc *<long>* (\textsmaller{<short>}) acronym style that has an accompanying description (which the user needs to supply).

```

10226 \renewacronymstyle{sm-short-long-desc}%
10227 {%
10228   \GlsUseAcrEntryDispStyle{sm-short-long}%
10229 }%
10230 {%
10231   \GlsUseAcrStyleDefs{sm-short-long}%
10232   \renewcommand*{\GenericAcronymFields}{}%
10233   \renewcommand*{\acronymsort}[2]{##2}%
10234   \renewcommand*{\acronymentry}[1]{%
10235     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10236     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10237 }
```

dua *<long>* only acronym style.

```

10238 \renewacronymstyle{dua}%
10239 {%
```

Check for long form in case this is a mixed glossary.

```

10240 \ifdefempty\glscustomtext
10241 {%
10242   \ifglshaslong{\glslabel}%
10243   {%
10244     \glslifplural
10245     {%
```

Plural form:

```

10246     \glscapscase
10247     {%
```

Plural form, don't adjust case:

```

10248     \glslongpluralaccessdisplay{\glsentrylongpl{\glslabel}}{\glslabel}%
10249     \glsinsert
10250   }%
10251   {%
```

Plural form, make first letter upper case:

```

10252     \glslongpluralaccessdisplay{\Glsentrylongpl{\glslabel}}{\glslabel}%
10253     \glsinsert
10254   }%
10255   {%
```

Plural form, all caps:

```

10256     \glslongpluralaccessdisplay
10257     {\mfirstucMakeUppercase{\glsentrylongpl{\glslabel}}}{\glslabel}%
10258     \mfirstucMakeUppercase{\glsinsert}%
10259   }%
10260   }%
10261   {%
```

Singular form

```
10262      \glscapscase
10263      {%
```

Singular form, don't adjust case:

```
10264      \glslongaccessdisplay{\glsentrylong{\glslabel}}{\glslabel}\glsinsert
10265      }%
10266      {%
```

Subsequent singular form, make first letter upper case:

```
10267      \glslongaccessdisplay{\Glsentrylong{\glslabel}}{\glslabel}\glsinsert
10268      }%
10269      {%
```

Subsequent singular form, all caps:

```
10270      \glslongaccessdisplay
10271      {\mfirstucMakeUppercase
10272      {\glsentrylong{\glslabel}\glsinsert}}{\glslabel}%
10273      \mfirstucMakeUppercase{\glsinsert}%
10274      }%
10275      }%
10276      }%
10277      {%
```

Not an acronym:

```
10278      \glsgenentryfmt
10279      }%
10280      }%
10281      {\glscustomtext\glsinsert}%
10282      }%
10283      {%
10284      \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
10285      \renewcommand*{\acrfullfmt}[3]{%
10286      \glslink[##1]{##2}{%
10287      \glslongaccessdisplay{\glsentrylong{##2}}{##2}##3\space
10288      (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##2}}{##2}})}%
10289      \renewcommand*{\Acrfullfmt}[3]{%
10290      \glslink[##1]{##2}{%
10291      \glslongaccessdisplay{\Glsentrylong{##2}}{##2}##3\space
10292      (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##2}}{##2}})}%
10293      \renewcommand*{\ACRfullfmt}[3]{%
10294      \glslink[##1]{##2}{%
10295      \glslongaccessdisplay
10296      {\mfirstucMakeUppercase{\glsentrylong{##2}}{##2}##3\space
10297      (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##2}}{##2}})}%
10298      \renewcommand*{\acrfullplfmt}[3]{%
10299      \glslink[##1]{##2}{%
10300      \glslongpluralaccessdisplay
10301      {\glsentrylongpl{##2}}{##2}##3\space
10302      (\glsshortpluralaccessdisplay
10303      {\acronymfont{\glsentryshortpl{##2}}{##2}})}%
```

```

10304 \renewcommand*{\Acrfullplfmt}[3]{%
10305     \glslink[##1]{##2}{%
10306         \glslongpluralaccessdisplay
10307         {\Glsentrylongpl{##2}}{##2}##3\space
10308         (\glsshortpluralaccessdisplay
10309         {\acronymfont{\glsentryshortpl{##2}}{##2}})}%
10310 \renewcommand*{\ACRfullplfmt}[3]{%
10311     \glslink[##1]{##2}{%
10312         \glslongpluralaccessdisplay
10313         {\mfirstucMakeUppercase{\glsentrylongpl{##2}}{##2}##3\space
10314         (\glsshortpluralaccessdisplay
10315         {\acronymfont{\glsentryshortpl{##2}}{##2}})}%
10316 \renewcommand*{\glsentryfull}[1]{%
10317     \glslongaccessdisplay{\glsentrylong{##1}}\space
10318     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}{##1}})%
10319 }%
10320 \renewcommand*{\Glsentryfull}[1]{%
10321     \glslongaccessdisplay{\Glsentrylong{##1}}{##1}\space
10322     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}{##1}})%
10323 }%
10324 \renewcommand*{\glsentryfullpl}[1]{%
10325     \glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1}\space
10326     (\glsshortpluralaccessdisplay{\acronymfont{\glsentryshortpl{##1}}{##1}})%
10327 }%
10328 \renewcommand*{\Glsentryfullpl}[1]{%
10329     \glslongpluralaccessdisplay{\Glsentrylongpl{##1}}{##1}\space
10330     (\glsshortpluralaccessdisplay{\acronymfont{\glsentryshortpl{##1}}{##1}})%
10331 }%
10332 \renewcommand*{\acronymentry}[1]{%
10333     \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}{##1}}%
10334 \renewcommand*{\acronymsort}[2]{##1}%
10335 \renewcommand*{\acronymfont}[1]{##1}%
10336 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
10337 }

```

dua-desc *<long>* only acronym style with user-supplied description.

```

10338 \renewacronymstyle{dua-desc}%
10339 {%
10340     \GlsUseAcrEntryDispStyle{dua}%
10341 }%
10342 {%
10343     \GlsUseAcrStyleDefs{dua}%
10344     \renewcommand*{\GenericAcronymFields}{}%
10345     \renewcommand*{\acronymentry}[1]{%
10346         \glslongaccessdisplay{\acronymfont{\glsentrylong{##1}}{##1}}%
10347     \renewcommand*{\acronymsort}[2]{##2}%
10348 }%

```

footnote *<short>*\footnote{*<long>*} acronym style.

10349 \renewacronymstyle{footnote}%

10350 {%

Check for long form in case this is a mixed glossary.

10351 \ifglshaslong{\glslabel}{\glsngenacfmt}{\glsngenentryfmt}%

10352 }%

10353 {%

10354 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%

Need to ensure hyperlinks are switched off on first use:

10355 \glshyperfirstfalse

10356 \renewcommand*{\genacrfullformat}[2]{%

10357 \glsshortaccessdisplay

10358 {\protect\firstacronymfont{\glsentryshort{##1}}}{##1}##2%

10359 \protect\footnote{\glslongaccessdisplay{\glsentrylong{##1}}}{##1}}%

10360 }%

10361 \renewcommand*{\Genacrfullformat}[2]{%

10362 \glsshortaccessdisplay

10363 {\firstacronymfont{\Glsentryshort{##1}}}{##1}##2%

10364 \protect\footnote{\glslongaccessdisplay{\glsentrylong{##1}}}{##1}}%

10365 }%

10366 \renewcommand*{\genplacrfullformat}[2]{%

10367 \glsshortpluralaccessdisplay

10368 {\protect\firstacronymfont{\glsentryshortpl{##1}}}{##1}##2%

10369 \protect\footnote{\glslongpluralaccessdisplay{\glsentrylongpl{##1}}}{##1}}%

10370 }%

10371 \renewcommand*{\Genplacrfullformat}[2]{%

10372 \glsshortpluralaccessdisplay

10373 {\protect\firstacronymfont{\Glsentryshortpl{##1}}}{##1}##2%

10374 \protect\footnote{\glslongpluralaccessdisplay{\glsentrylongpl{##1}}}{##1}}%

10375 }%

10376 \renewcommand*{\acronymentry}[1]{%

10377 \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}}%

10378 \renewcommand*{\acronymsort}[2]{##1}%

10379 \renewcommand*{\acronymfont}[1]{##1}%

10380 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%

Don't use footnotes for \acrfull:

10381 \renewcommand*{\acrfullfmt}[3]{%

10382 \glslink[##1]{##2}{%

10383 \glsshortaccessdisplay{\acronymfont{\glsentryshort{##2}}}{##2}##3\space

10384 (\glslongaccessdisplay{\glsentrylong{##2}}{##2}})%

10385 \renewcommand*{\Acrfullfmt}[3]{%

10386 \glslink[##1]{##2}{%

10387 \glsshortaccessdisplay{\acronymfont{\Glsentryshort{##2}}}{##2}##3\space

10388 (\glslongaccessdisplay{\glsentrylong{##2}}{##2}})%

10389 \renewcommand*{\ACRfullfmt}[3]{%

10390 \glslink[##1]{##2}{%

10391 \glsshortaccessdisplay

10392 {\mfirstucMakeUppercase

10393 {\acronymfont{\glsentryshort{##2}}}{##2}##3\space

```

10394      (\glslongaccessdisplay{\glsentrylong{##2}}{##2}})}%
10395 \renewcommand*{\acrfullplfmt}[3]{%
10396   \glslink[##1]{##2}{%
10397     \glsshortpluralaccessdisplay
10398     {\acronymfont{\glsentryshortpl{##2}}{##2}##3\space
10399     (\glslongpluralaccessdisplay{\glsentrylongpl{##2}}{##2}})}%
10400 \renewcommand*{\Acrfullplfmt}[3]{%
10401   \glslink[##1]{##2}{%
10402     \glsshortpluralaccessdisplay
10403     {\acronymfont{\Glsentryshortpl{##2}}{##2}##3\space
10404     (\glslongpluralaccessdisplay{\glsentrylongpl{##2}})}%
10405 \renewcommand*{\ACRfullplfmt}[3]{%
10406   \glslink[##1]{##2}{%
10407     \glsshortpluralaccessdisplay
10408     {\mfirstucMakeUppercase
10409     {\acronymfont{\glsentryshortpl{##2}}{##2}##3\space
10410     (\glslongpluralaccessdisplay{\glsentrylongpl{##2}}{##2}})}%

```

Similarly for \glsentryfull etc:

```

10411 \renewcommand*{\glsentryfull}[1]{%
10412   \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}{##1}\space
10413   (\glslongaccessdisplay{\glsentrylong{##1}}{##1}})%
10414 \renewcommand*{\Glsentryfull}[1]{%
10415   \glsshortaccessdisplay{\acronymfont{\Glsentryshort{##1}}{##1}\space
10416   (\glslongaccessdisplay{\glsentrylong{##1}}{##1}})%
10417 \renewcommand*{\glsentryfullpl}[1]{%
10418   \glsshortpluralaccessdisplay
10419   {\acronymfont{\glsentryshortpl{##1}}{##1}\space
10420   (\glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1}})%
10421 \renewcommand*{\Glsentryfullpl}[1]{%
10422   \glsshortpluralaccessdisplay
10423   {\acronymfont{\Glsentryshortpl{##1}}{##1}\space
10424   (\glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1}})%
10425 }

```

footnote-sc \textsc{<short>}\footnote{<long>} acronym style.

```

10426 \renewacronymstyle{footnote-sc}%
10427 {%
10428   \GlsUseAcrEntryDisplayStyle{footnote}%
10429 }%
10430 {%
10431   \GlsUseAcrStyleDefs{footnote}%
10432   \renewcommand{\acronymentry}[1]{%
10433     \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}{##1}}
10434   \renewcommand{\acronymfont}[1]{\textsc{##1}}%
10435   \renewcommand*{\acrpluralsuffix}{\glstextup{\glspluralsuffix}}%
10436 }%

```

footnote-sm \textsmaller{<short>}\footnote{<long>} acronym style.

```

10437 \renewacronymstyle{footnote-sm}%
10438 {%
10439   \GlsUseAcrEntryDispStyle{footnote}%
10440 }%
10441 {%
10442   \GlsUseAcrStyleDefs{footnote}%
10443   \renewcommand{\acronymentry}[1]{%
10444     \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}}
10445   \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
10446   \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
10447 }%

```

footnote-desc *<short>*\footnote{*<long>*} acronym style that has an accompanying description (which the user needs to supply).

```

10448 \renewacronymstyle{footnote-desc}%
10449 {%
10450   \GlsUseAcrEntryDispStyle{footnote}%
10451 }%
10452 {%
10453   \GlsUseAcrStyleDefs{footnote}%
10454   \renewcommand*{\GenericAcronymFields}{}%
10455   \renewcommand*{\acronymsort}[2]{##2}%
10456   \renewcommand*{\acronymentry}[1]{%
10457     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10458     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1))}%
10459 }

```

footnote-sc-desc \textsc{*<short>*}\footnote{*<long>*} acronym style that has an accompanying description (which the user needs to supply).

```

10460 \renewacronymstyle{footnote-sc-desc}%
10461 {%
10462   \GlsUseAcrEntryDispStyle{footnote-sc}%
10463 }%
10464 {%
10465   \GlsUseAcrStyleDefs{footnote-sc}%
10466   \renewcommand*{\GenericAcronymFields}{}%
10467   \renewcommand*{\acronymsort}[2]{##2}%
10468   \renewcommand*{\acronymentry}[1]{%
10469     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10470     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1))}%
10471 }

```

footnote-sm-desc \textsmaller{*<short>*}\footnote{*<long>*} acronym style that has an accompanying description (which the user needs to supply).

```

10472 \renewacronymstyle{footnote-sm-desc}%
10473 {%
10474   \GlsUseAcrEntryDispStyle{footnote-sm}%
10475 }%
10476 {%

```

```

10477 \GlsUseAcrStyleDefs{footnote-sm}%
10478 \renewcommand*{\GenericAcronymFields}{}%
10479 \renewcommand*{\acronymsort}[2]{##2}%
10480 \renewcommand*{\acronymentry}[1]{%
10481   \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10482   (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10483 }

```

Use `\newacronymhook` to modify the key list to set the access text to the long version by default.

```

10484 \renewcommand*{\newacronymhook}{%
10485   \edef\@gls@keylist{shortaccess=\the\glslongtok,%
10486     \the\glskeylisttok}%
10487   \expandafter\glskeylisttok\expandafter{\@gls@keylist}%
10488 }

```

`\defaultNewAcronymDef` Modify default style to use access text:

```

10489 \renewcommand*{\DefaultNewAcronymDef}{%
10490   \edef\@do@newglossaryentry{%
10491     \noexpand\newglossaryentry{\the\glslabeltok}%
10492     {%
10493       type=\acronymtype,%
10494       name={\the\glsshorttok},%
10495       description={\the\glslongtok},%
10496       descriptionaccess=\relax,%
10497       text={\the\glsshorttok},%
10498       access={\noexpand\@glo@textaccess},%
10499       sort={\the\glsshorttok},%
10500       short={\the\glsshorttok},%
10501       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10502       shortaccess={\the\glslongtok},%
10503       long={\the\glslongtok},%
10504       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10505       descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10506       first={\noexpand\glslongaccessdisplay
10507         {\the\glslongtok}{\the\glslabeltok}\space
10508         (\noexpand\glsshortaccessdisplay
10509           {\the\glsshorttok}{\the\glslabeltok})},%
10510       plural={\the\glsshorttok\acrpluralsuffix},%
10511       firstplural={\noexpand\glslongpluralaccessdisplay
10512         {\noexpand\@glo@longpl}{\the\glslabeltok}\space
10513         (\noexpand\glsshortpluralaccessdisplay
10514           {\noexpand\@glo@shortpl}{\the\glslabeltok})},%
10515       firstaccess=\relax,%
10516       firstpluralaccess=\relax,%
10517       textaccess={\noexpand\@glo@shortaccess},%
10518       \the\glskeylisttok
10519     }%
10520   }%

```

```

10521 \let\@org@gl@s@assign@firstpl\gl@s@assign@firstpl
10522 \let\@org@gl@s@assign@plural\gl@s@assign@plural
10523 \let\@org@gl@s@assign@descplural\gl@s@assign@descplural
10524 \def\gl@s@assign@firstpl##1##2{%
10525   \@@gl@s@expand@field{##1}{firstpl}{##2}%
10526 }%
10527 \def\gl@s@assign@plural##1##2{%
10528   \@@gl@s@expand@field{##1}{plural}{##2}%
10529 }%
10530 \def\gl@s@assign@descplural##1##2{%
10531   \@@gl@s@expand@field{##1}{descplural}{##2}%
10532 }%
10533 \do@newglossaryentry
10534 \let\gl@s@assign@firstpl\@org@gl@s@assign@firstpl
10535 \let\gl@s@assign@plural\@org@gl@s@assign@plural
10536 \let\gl@s@assign@symbolplural\@org@gl@s@assign@symbolplural
10537 }

```

otnoteNewAcronymDef

```

10538 \renewcommand*{\DescriptionFootnoteNewAcronymDef}{%
10539   \edef\@do@newglossaryentry{%
10540     \noexpand\newglossaryentry{\the\glslabelltok}%
10541     {%
10542       type=\acronymtype,%
10543       name={\noexpand\acronymfont{\the\glsshorttok}},%
10544       sort={\the\glsshorttok},%
10545       text={\the\glsshorttok},%
10546       short={\the\glsshorttok},%
10547       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10548       shortaccess={\the\glslongtok},%
10549       long={\the\glslongtok},%
10550       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10551       access={\noexpand\@glo@textaccess},%
10552       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10553       symbol={\the\glslongtok},%
10554       symbolplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10555       firstpluralaccess=\relax,
10556       textaccess={\noexpand\@glo@shortaccess},%
10557       \the\glscopylisttok
10558     }%
10559   }%
10560 \let\@org@gl@s@assign@firstpl\gl@s@assign@firstpl
10561 \let\@org@gl@s@assign@plural\gl@s@assign@plural
10562 \let\@org@gl@s@assign@symbolplural\gl@s@assign@symbolplural
10563 \def\gl@s@assign@firstpl##1##2{%
10564   \@@gl@s@expand@field{##1}{firstpl}{##2}%
10565 }%
10566 \def\gl@s@assign@plural##1##2{%
10567   \@@gl@s@expand@field{##1}{plural}{##2}%

```

```

10568 }%
10569 \def\gls@assign@symbolplural##1##2{%
10570   @@gls@expand@field{##1}{symbolplural}{##2}%
10571 }%
10572 \@do@newglossaryentry
10573 \let\gls@assign@plural\@org@gls@assign@plural
10574 \let\gls@assign@firstpl\@org@gls@assign@firstpl
10575 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
10576 }

```

ptionNewAcronymDef

```

10577 \renewcommand*{\DescriptionNewAcronymDef}{%
10578   \edef\@do@newglossaryentry{%
10579     \noexpand\newglossaryentry{\the\glslabeltok}%
10580     {%
10581       type=\acronymtype,%
10582       name={\noexpand
10583         \acrnameformat{\the\glsshorttok}{\the\glslongtok}},%
10584       access={\noexpand\@glo@textaccess},%
10585       sort={\the\glsshorttok},%
10586       short={\the\glsshorttok},%
10587       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10588       shortaccess={\the\glslongtok},%
10589       long={\the\glslongtok},%
10590       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10591       first={\the\glslongtok},%
10592       firstaccess=\relax,
10593       firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10594       text={\the\glsshorttok},%
10595       textaccess={\the\glslongtok},%
10596       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10597       symbol={\noexpand\@glo@text},%
10598       symbolaccess={\noexpand\@glo@textaccess},%
10599       symbolplural={\noexpand\@glo@plural},%
10600       firstpluralaccess=\relax,
10601       textaccess={\noexpand\@glo@shortaccess},%
10602       \the\glskeylisttok}%
10603   }%
10604   \let\@org@gls@assign@firstpl\gls@assign@firstpl
10605   \let\@org@gls@assign@plural\gls@assign@plural
10606   \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
10607   \def\gls@assign@firstpl##1##2{%
10608     @@gls@expand@field{##1}{firstpl}{##2}%
10609   }%
10610   \def\gls@assign@plural##1##2{%
10611     @@gls@expand@field{##1}{plural}{##2}%
10612   }%
10613   \def\gls@assign@symbolplural##1##2{%
10614     @@gls@expand@field{##1}{symbolplural}{##2}%

```

```

10615 }%
10616 \@do@newglossaryentry
10617 \let\gls@assign@firstpl\@org@gls@assign@firstpl
10618 \let\gls@assign@plural\@org@gls@assign@plural
10619 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
10620 }

```

otnoteNewAcronymDef

```

10621 \renewcommand*{\FootnoteNewAcronymDef}{%
10622   \edef\@do@newglossaryentry{%
10623     \noexpand\newglossaryentry{\the\glslabeltok}%
10624     {%
10625       type=\acronymtype,%
10626       name={\noexpand\acronymfont{\the\glsshorttok}},%
10627       sort={\the\glsshorttok},%
10628       text={\the\glsshorttok},%
10629       textaccess={\the\glslongtok},%
10630       access={\noexpand\@glo@textaccess},%
10631       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10632       short={\the\glsshorttok},%
10633       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10634       long={\the\glslongtok},%
10635       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10636       description={\the\glslongtok},%
10637       descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10638       \the\glskeylisttok
10639     }%
10640   }%
10641   \let\@org@gls@assign@plural\gls@assign@plural
10642   \let\@org@gls@assign@firstpl\gls@assign@firstpl
10643   \let\@org@gls@assign@descplural\gls@assign@descplural
10644   \def\gls@assign@firstpl##1##2{%
10645     \@@gls@expand@field{##1}{firstpl}{##2}%
10646   }%
10647   \def\gls@assign@plural##1##2{%
10648     \@@gls@expand@field{##1}{plural}{##2}%
10649   }%
10650   \def\gls@assign@descplural##1##2{%
10651     \@@gls@expand@field{##1}{descplural}{##2}%
10652   }%
10653   \@do@newglossaryentry
10654   \let\gls@assign@plural\@org@gls@assign@plural
10655   \let\gls@assign@firstpl\@org@gls@assign@firstpl
10656   \let\gls@assign@descplural\@org@gls@assign@descplural
10657 }

```

\SmallNewAcronymDef

```

10658 \renewcommand*{\SmallNewAcronymDef}{%
10659   \edef\@do@newglossaryentry{%

```

```

10660 \noexpand\newglossaryentry{\the\glslabeltok}%
10661 {%
10662     type=\acronymtype,%
10663     name={\noexpand\acronymfont{\the\glsshorttok}},%
10664     access={\noexpand\@glo@symbolaccess},%
10665     sort={\the\glsshorttok},%
10666     short={\the\glsshorttok},%
10667     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10668     shortaccess={\the\glslongtok},%
10669     long={\the\glslongtok},%
10670     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10671     text={\noexpand\@glo@short},%
10672     textaccess={\noexpand\@glo@shortaccess},%
10673     plural={\noexpand\@glo@shortpl},%
10674     first={\the\glslongtok},%
10675     firstaccess=\relax,%
10676     firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10677     description={\noexpand\@glo@first},%
10678     descriptionplural={\noexpand\@glo@firstplural},%
10679     symbol={\the\glsshorttok},%
10680     symbolaccess={\the\glslongtok},%
10681     symbolplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10682     \the\glskeylisttok
10683 }%
10684 }%
10685 \let\@org@gls@assign@firstpl\gls@assign@firstpl
10686 \let\@org@gls@assign@plural\gls@assign@plural
10687 \let\@org@gls@assign@descplural\gls@assign@descplural
10688 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
10689 \def\gls@assign@firstpl##1##2{%
10690     \@@gls@expand@field{##1}{firstpl}{##2}%
10691 }%
10692 \def\gls@assign@plural##1##2{%
10693     \@@gls@expand@field{##1}{plural}{##2}%
10694 }%
10695 \def\gls@assign@descplural##1##2{%
10696     \@@gls@expand@field{##1}{descplural}{##2}%
10697 }%
10698 \def\gls@assign@symbolplural##1##2{%
10699     \@@gls@expand@field{##1}{symbolplural}{##2}%
10700 }%
10701 \@do@newglossaryentry
10702 \let\gls@assign@firstpl\@org@gls@assign@firstpl
10703 \let\gls@assign@plural\@org@gls@assign@plural
10704 \let\gls@assign@descplural\@org@gls@assign@descplural
10705 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
10706 }

```

The following are kept for compatibility with versions before 3.0:


```

\glsshortaccesskey
10707 \newcommand*{\glsshortaccesskey}{\glsshortkey access}%

hortpluralaccesskey
10708 \newcommand*{\glsshortpluralaccesskey}{\glsshortpluralkey access}%

\glslongaccesskey
10709 \newcommand*{\glslongaccesskey}{\glslongkey access}%

longpluralaccesskey
10710 \newcommand*{\glslongpluralaccesskey}{\glslongpluralkey access}%

```

6.5 Debugging Commands

```

\showglonameaccess
10711 \newcommand*{\showglonameaccess}[1]{%
10712 \expandafter\show\csname glo@\glsdetoklabel{#1}@textaccess\endcsname
10713 }

\showglotextaccess
10714 \newcommand*{\showglotextaccess}[1]{%
10715 \expandafter\show\csname glo@\glsdetoklabel{#1}@textaccess\endcsname
10716 }

showglopluralaccess
10717 \newcommand*{\showglopluralaccess}[1]{%
10718 \expandafter\show\csname glo@\glsdetoklabel{#1}@pluralaccess\endcsname
10719 }

\showglofirstaccess
10720 \newcommand*{\showglofirstaccess}[1]{%
10721 \expandafter\show\csname glo@\glsdetoklabel{#1}@firstaccess\endcsname
10722 }

lofirstpluralaccess
10723 \newcommand*{\showglofirstpluralaccess}[1]{%
10724 \expandafter\show\csname glo@\glsdetoklabel{#1}@firstpluralaccess\endcsname
10725 }

showglosymbolaccess
10726 \newcommand*{\showglosymbolaccess}[1]{%
10727 \expandafter\show\csname glo@\glsdetoklabel{#1}@symbolaccess\endcsname
10728 }

osymbolpluralaccess
10729 \newcommand*{\showglosymbolpluralaccess}[1]{%
10730 \expandafter\show\csname glo@\glsdetoklabel{#1}@symbolpluralaccess\endcsname
10731 }

```

```

\showglodescaccess
10732 \newcommand*{\showglodescaccess}[1]{%
10733   \expandafter\show\csname glo@\glsdetoklabel{#1}@descaccess\endcsname
10734 }

glodescpluralaccess
10735 \newcommand*{\showglodescpluralaccess}[1]{%
10736   \expandafter\show\csname glo@\glsdetoklabel{#1}@descpluralaccess\endcsname
10737 }

\showgloshortaccess
10738 \newcommand*{\showgloshortaccess}[1]{%
10739   \expandafter\show\csname glo@\glsdetoklabel{#1}@shortaccess\endcsname
10740 }

loshortpluralaccess
10741 \newcommand*{\showgloshortpluralaccess}[1]{%
10742   \expandafter\show\csname glo@\glsdetoklabel{#1}@shortpluralaccess\endcsname
10743 }

\showglolongaccess
10744 \newcommand*{\showglolongaccess}[1]{%
10745   \expandafter\show\csname glo@\glsdetoklabel{#1}@longaccess\endcsname
10746 }

glolongpluralaccess
10747 \newcommand*{\showglolongpluralaccess}[1]{%
10748   \expandafter\show\csname glo@\glsdetoklabel{#1}@longpluralaccess\endcsname
10749 }

```

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.

```

10750 \NeedsTeXFormat{LaTeX2e}
10751 \ProvidesPackage{glossaries-babel}[2013/11/14 v4.0 (NLCT)]

English:
10752 \@ifundefined{captionsenglish}{}{%
10753   \addto\captionsenglish{%
10754     \renewcommand*{\glossaryname}{Glossary}%
10755     \renewcommand*{\acronymname}{Acronyms}%

```

```

10756 \renewcommand*{\entryname}{Notation}%
10757 \renewcommand*{\descriptionname}{Description}%
10758 \renewcommand*{\symbolname}{Symbol}%
10759 \renewcommand*{\pagelistname}{Page List}%
10760 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10761 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10762 }%
10763 }
10764 \@ifundefined{captionsamerican}{}{%
10765 \addto\captionsamerican{%
10766 \renewcommand*{\glossaryname}{Glossary}%
10767 \renewcommand*{\acronymname}{Acronyms}%
10768 \renewcommand*{\entryname}{Notation}%
10769 \renewcommand*{\descriptionname}{Description}%
10770 \renewcommand*{\symbolname}{Symbol}%
10771 \renewcommand*{\pagelistname}{Page List}%
10772 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10773 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10774 }%
10775 }
10776 \@ifundefined{captionsaustralian}{}{%
10777 \addto\captionsaustralian{%
10778 \renewcommand*{\glossaryname}{Glossary}%
10779 \renewcommand*{\acronymname}{Acronyms}%
10780 \renewcommand*{\entryname}{Notation}%
10781 \renewcommand*{\descriptionname}{Description}%
10782 \renewcommand*{\symbolname}{Symbol}%
10783 \renewcommand*{\pagelistname}{Page List}%
10784 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10785 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10786 }%
10787 }
10788 \@ifundefined{captionsbritish}{}{%
10789 \addto\captionsbritish{%
10790 \renewcommand*{\glossaryname}{Glossary}%
10791 \renewcommand*{\acronymname}{Acronyms}%
10792 \renewcommand*{\entryname}{Notation}%
10793 \renewcommand*{\descriptionname}{Description}%
10794 \renewcommand*{\symbolname}{Symbol}%
10795 \renewcommand*{\pagelistname}{Page List}%
10796 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10797 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10798 }%
10799 \@ifundefined{captionscanadian}{}{%
10800 \addto\captionscanadian{%
10801 \renewcommand*{\glossaryname}{Glossary}%
10802 \renewcommand*{\acronymname}{Acronyms}%
10803 \renewcommand*{\entryname}{Notation}%
10804 \renewcommand*{\descriptionname}{Description}%

```

```

10805 \renewcommand*{\symbolname}{Symbol}%
10806 \renewcommand*{\pagelistname}{Page List}%
10807 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10808 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10809 }%
10810 }
10811 \@ifundefined{captionsnewzealand}{}{%
10812 \addto\captionsnewzealand{%
10813 \renewcommand*{\glossaryname}{Glossary}%
10814 \renewcommand*{\acronymname}{Acronyms}%
10815 \renewcommand*{\entryname}{Notation}%
10816 \renewcommand*{\descriptionname}{Description}%
10817 \renewcommand*{\symbolname}{Symbol}%
10818 \renewcommand*{\pagelistname}{Page List}%
10819 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10820 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10821 }%
10822 }
10823 \@ifundefined{captionsUKenglish}{}{%
10824 \addto\captionsUKenglish{%
10825 \renewcommand*{\glossaryname}{Glossary}%
10826 \renewcommand*{\acronymname}{Acronyms}%
10827 \renewcommand*{\entryname}{Notation}%
10828 \renewcommand*{\descriptionname}{Description}%
10829 \renewcommand*{\symbolname}{Symbol}%
10830 \renewcommand*{\pagelistname}{Page List}%
10831 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10832 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10833 }%
10834 }
10835 \@ifundefined{captionsUSenglish}{}{%
10836 \addto\captionsUSenglish{%
10837 \renewcommand*{\glossaryname}{Glossary}%
10838 \renewcommand*{\acronymname}{Acronyms}%
10839 \renewcommand*{\entryname}{Notation}%
10840 \renewcommand*{\descriptionname}{Description}%
10841 \renewcommand*{\symbolname}{Symbol}%
10842 \renewcommand*{\pagelistname}{Page List}%
10843 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10844 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10845 }%
10846 }

```

German (quite a few variations were suggested for German; I settled on the following):

```

10847 \@ifundefined{captionsgerman}{}{%
10848 \addto\captionsgerman{%
10849 \renewcommand*{\glossaryname}{Glossar}%
10850 \renewcommand*{\acronymname}{Akronyme}%
10851 \renewcommand*{\entryname}{Bezeichnung}%

```

```

10852 \renewcommand*{\descriptionname}{Beschreibung}%
10853 \renewcommand*{\symbolname}{Symbol}%
10854 \renewcommand*{\pagelistname}{Seiten}%
10855 \renewcommand*{\glssymbolsgroupname}{Symbole}%
10856 \renewcommand*{\glsnumbersgroupname}{Zahlen}}
10857 }

```

ngerman is identical to German:

```

10858 \@ifundefined{captionsngerman}{}{%
10859 \addto\captionsngerman{%
10860 \renewcommand*{\glossaryname}{Glossar}%
10861 \renewcommand*{\acronymname}{Akronyme}%
10862 \renewcommand*{\entryname}{Bezeichnung}%
10863 \renewcommand*{\descriptionname}{Beschreibung}%
10864 \renewcommand*{\symbolname}{Symbol}%
10865 \renewcommand*{\pagelistname}{Seiten}%
10866 \renewcommand*{\glssymbolsgroupname}{Symbole}%
10867 \renewcommand*{\glsnumbersgroupname}{Zahlen}}
10868 }

```

Italian:

```

10869 \@ifundefined{captionsitalian}{}{%
10870 \addto\captionsitalian{%
10871 \renewcommand*{\glossaryname}{Glossario}%
10872 \renewcommand*{\acronymname}{Acronimi}%
10873 \renewcommand*{\entryname}{Nomenclatura}%
10874 \renewcommand*{\descriptionname}{Descrizione}%
10875 \renewcommand*{\symbolname}{Simbolo}%
10876 \renewcommand*{\pagelistname}{Elenco delle pagine}%
10877 \renewcommand*{\glssymbolsgroupname}{Simboli}%
10878 \renewcommand*{\glsnumbersgroupname}{Numeri}}
10879 }

```

Dutch:

```

10880 \@ifundefined{captionsdutch}{}{%
10881 \addto\captionsdutch{%
10882 \renewcommand*{\glossaryname}{Woordenlijst}%
10883 \renewcommand*{\acronymname}{Acroniemen}%
10884 \renewcommand*{\entryname}{Benaming}%
10885 \renewcommand*{\descriptionname}{Beschrijving}%
10886 \renewcommand*{\symbolname}{Symbool}%
10887 \renewcommand*{\pagelistname}{Pagina's}%
10888 \renewcommand*{\glssymbolsgroupname}{Symbolen}%
10889 \renewcommand*{\glsnumbersgroupname}{Cijfers}}
10890 }

```

Spanish:

```

10891 \@ifundefined{captionsspanish}{}{%
10892 \addto\captionsspanish{%
10893 \renewcommand*{\glossaryname}{Glosario}%
10894 \renewcommand*{\acronymname}{Siglas}%

```

```

10895 \renewcommand*{\entryname}{Entrada}%
10896 \renewcommand*{\descriptionname}{Descripci\`on}%
10897 \renewcommand*{\symbolname}{S\`{\i}mbolo}%
10898 \renewcommand*{\pagelistname}{Lista de p\`aginas}%
10899 \renewcommand*{\glssymbolsgroupname}{S\`{\i}mbolos}%
10900 \renewcommand*{\glslnumbersgroupname}{N\`umeros}}
10901 }

```

French:

```

10902 \@ifundefined{captionsfrench}{}{%
10903 \addto\captionsfrench{%
10904 \renewcommand*{\glossaryname}{Glossaire}%
10905 \renewcommand*{\acronymname}{Acronymes}%
10906 \renewcommand*{\entryname}{Terme}%
10907 \renewcommand*{\descriptionname}{Description}%
10908 \renewcommand*{\symbolname}{Symbole}%
10909 \renewcommand*{\pagelistname}{Pages}%
10910 \renewcommand*{\glssymbolsgroupname}{Symboles}%
10911 \renewcommand*{\glslnumbersgroupname}{Nombres}}
10912 }
10913 \@ifundefined{captionsfrenchb}{}{%
10914 \addto\captionsfrenchb{%
10915 \renewcommand*{\glossaryname}{Glossaire}%
10916 \renewcommand*{\acronymname}{Acronymes}%
10917 \renewcommand*{\entryname}{Terme}%
10918 \renewcommand*{\descriptionname}{Description}%
10919 \renewcommand*{\symbolname}{Symbole}%
10920 \renewcommand*{\pagelistname}{Pages}%
10921 \renewcommand*{\glssymbolsgroupname}{Symboles}%
10922 \renewcommand*{\glslnumbersgroupname}{Nombres}}
10923 }
10924 \@ifundefined{captionsfrancais}{}{%
10925 \addto\captionsfrancais{%
10926 \renewcommand*{\glossaryname}{Glossaire}%
10927 \renewcommand*{\acronymname}{Acronymes}%
10928 \renewcommand*{\entryname}{Terme}%
10929 \renewcommand*{\descriptionname}{Description}%
10930 \renewcommand*{\symbolname}{Symbole}%
10931 \renewcommand*{\pagelistname}{Pages}%
10932 \renewcommand*{\glssymbolsgroupname}{Symboles}%
10933 \renewcommand*{\glslnumbersgroupname}{Nombres}}
10934 }

```

Danish:

```

10935 \@ifundefined{captionsdanish}{}{%
10936 \addto\captionsdanish{%
10937 \renewcommand*{\glossaryname}{Ordliste}%
10938 \renewcommand*{\acronymname}{Akronymer}%
10939 \renewcommand*{\entryname}{Symbolforklaring}%
10940 \renewcommand*{\descriptionname}{Beskrivelse}%

```

```

10941 \renewcommand*{\symbolname}{Symbol}%
10942 \renewcommand*{\pagelistname}{Side}%
10943 \renewcommand*{\glssymbolsgroupname}{Symboler}%
10944 \renewcommand*{\glsnnumbersgroupname}{Tal}}
10945 }

```

Irish:

```

10946 \@ifundefined{captionsirish}{}{%
10947 \addto\captionsirish{%
10948 \renewcommand*{\glossaryname}{Gluais}%
10949 \renewcommand*{\acronymname}{Acrainmneacha}%

```

wasn't sure whether to go for Nóta (Note), Ciall ('Meaning', 'sense') or Brí ('Meaning'). In the end I chose Ciall.

```

10950 \renewcommand*{\entryname}{Ciall}%
10951 \renewcommand*{\descriptionname}{Tuairisc}%

```

Again, not sure whether to use Comhartha/Comharthaí or Siombail/Siombaile, so have chosen the former.

```

10952 \renewcommand*{\symbolname}{Comhartha}%
10953 \renewcommand*{\glssymbolsgroupname}{Comhartha\'}{\i}}%
10954 \renewcommand*{\pagelistname}{Leathanaigh}%
10955 \renewcommand*{\glsnnumbersgroupname}{Uimhreacha}}
10956 }

```

Hungarian:

```

10957 \@ifundefined{captionsmagyar}{}{%
10958 \addto\captionsmagyar{%
10959 \renewcommand*{\glossaryname}{Sz\'ojegyz\'ek}%
10960 \renewcommand*{\acronymname}{Bet\H uszavak}%
10961 \renewcommand*{\entryname}{Kifejez\'es}%
10962 \renewcommand*{\descriptionname}{Magyar\'azat}%
10963 \renewcommand*{\symbolname}{Jel\'ol\'es}%
10964 \renewcommand*{\pagelistname}{Oldalsz\'am}%
10965 \renewcommand*{\glssymbolsgroupname}{Jelek}%
10966 \renewcommand*{\glsnnumbersgroupname}{Sz\'amjegyek}%
10967 }
10968 }
10969 \@ifundefined{captionshungarian}{}{%
10970 \addto\captionshungarian{%
10971 \renewcommand*{\glossaryname}{Sz\'ojegyz\'ek}%
10972 \renewcommand*{\acronymname}{Bet\H uszavak}%
10973 \renewcommand*{\entryname}{Kifejez\'es}%
10974 \renewcommand*{\descriptionname}{Magyar\'azat}%
10975 \renewcommand*{\symbolname}{Jel\'ol\'es}%
10976 \renewcommand*{\pagelistname}{Oldalsz\'am}%
10977 \renewcommand*{\glssymbolsgroupname}{Jelek}%
10978 \renewcommand*{\glsnnumbersgroupname}{Sz\'amjegyek}%
10979 }
10980 }

```

Polish

```

10981 \@ifundefined{captionspolish}{}{%
10982   \addto\captionspolish{%
10983     \renewcommand*{\glossaryname}{S{\l}ownik termin\'}ow}%
10984     \renewcommand*{\acronymname}{Skr\'}ot}%
10985     \renewcommand*{\entryname}{Termin}%
10986     \renewcommand*{\descriptionname}{Opis}%
10987     \renewcommand*{\symbolname}{Symbol}%
10988     \renewcommand*{\pagelistname}{Strony}%
10989     \renewcommand*{\glssymbolsgroupname}{Symbole}%
10990     \renewcommand*{\glsnumbersgroupname}{Liczby}}
10991 }

```

Brazilian

```

10992 \@ifundefined{captionsbrazil}{}{%
10993   \addto\captionsbrazil{%
10994     \renewcommand*{\glossaryname}{Gloss\'}ario}%
10995     \renewcommand*{\acronymname}{Siglas}%
10996     \renewcommand*{\entryname}{Nota\ c\ ~ao}%
10997     \renewcommand*{\descriptionname}{Descri\ c\ ~ao}%
10998     \renewcommand*{\symbolname}{S\'}imbolo}%
10999     \renewcommand*{\pagelistname}{Lista de P\'}aginas}%
11000     \renewcommand*{\glssymbolsgroupname}{S\'}imbolos}%
11001     \renewcommand*{\glsnumbersgroupname}{N\'}umeros}%
11002   }%
11003 }

```

7.2 Polyglossia Captions

```

11004 \NeedsTeXFormat{LaTeX2e}
11005 \ProvidesPackage{glossaries-polyglossia}[2013/11/14 v4.0 (NLCT)]

```

English:

```

11006 \@ifundefined{captionseenglish}{}{%
11007   \expandafter\toks@\expandafter{\captionseenglish
11008     \renewcommand*{\glossaryname}{\textenglish{Glossary}}}%
11009     \renewcommand*{\acronymname}{\textenglish{Acronyms}}}%
11010     \renewcommand*{\entryname}{\textenglish{Notation}}}%
11011     \renewcommand*{\descriptionname}{\textenglish{Description}}}%
11012     \renewcommand*{\symbolname}{\textenglish{Symbol}}}%
11013     \renewcommand*{\pagelistname}{\textenglish{Page List}}}%
11014     \renewcommand*{\glssymbolsgroupname}{\textenglish{Symbols}}}%
11015     \renewcommand*{\glsnumbersgroupname}{\textenglish{Numbers}}}%
11016   }%
11017   \edef\captionseenglish{\the\toks@}%
11018 }

```

German:

```

11019 \@ifundefined{captionsgerman}{}{%
11020   \expandafter\toks@\expandafter{\captionsgerman
11021     \renewcommand*{\glossaryname}{\textgerman{Glossar}}}%

```



```

11022 \renewcommand*{\acronymname}{\textgerman{Akronyme}}%
11023 \renewcommand*{\entryname}{\textgerman{Bezeichnung}}%
11024 \renewcommand*{\descriptionname}{\textgerman{Beschreibung}}%
11025 \renewcommand*{\symbolname}{\textgerman{Symbol}}%
11026 \renewcommand*{\pagelistname}{\textgerman{Seiten}}%
11027 \renewcommand*{\glssymbolsgroupname}{\textgerman{Symbole}}%
11028 \renewcommand*{\glsnumbersgroupname}{\textgerman{Zahlen}}%
11029 }%
11030 \edef\captionsgerman{\the\toks@}%
11031 }

```

Italian:

```

11032 \@ifundefined{captionsspanish}{\%
11033 \expandafter\toks@\expandafter{\captionsspanish
11034 \renewcommand*{\glossaryname}{\textitalian{Glossario}}%
11035 \renewcommand*{\acronymname}{\textitalian{Acronimi}}%
11036 \renewcommand*{\entryname}{\textitalian{Nomenclatura}}%
11037 \renewcommand*{\descriptionname}{\textitalian{Descrizione}}%
11038 \renewcommand*{\symbolname}{\textitalian{Simbolo}}%
11039 \renewcommand*{\pagelistname}{\textitalian{Elenco delle pagine}}%
11040 \renewcommand*{\glssymbolsgroupname}{\textitalian{Simboli}}%
11041 \renewcommand*{\glsnumbersgroupname}{\textitalian{Numeri}}%
11042 }%
11043 \edef\captionsspanish{\the\toks@}%
11044 }

```

Dutch:

```

11045 \@ifundefined{captionsdutch}{\%
11046 \expandafter\toks@\expandafter{\captionsdutch
11047 \renewcommand*{\glossaryname}{\textdutch{Woordenlijst}}%
11048 \renewcommand*{\acronymname}{\textdutch{Acroniemen}}%
11049 \renewcommand*{\entryname}{\textdutch{Benaming}}%
11050 \renewcommand*{\descriptionname}{\textdutch{Beschrijving}}%
11051 \renewcommand*{\symbolname}{\textdutch{Symbool}}%
11052 \renewcommand*{\pagelistname}{\textdutch{Pagina's}}%
11053 \renewcommand*{\glssymbolsgroupname}{\textdutch{Symbolen}}%
11054 \renewcommand*{\glsnumbersgroupname}{\textdutch{Cijfers}}%
11055 }%
11056 \edef\captionsdutch{\the\toks@}%
11057 }

```

Spanish:

```

11058 \@ifundefined{captionsspanish}{\%
11059 \expandafter\toks@\expandafter{\captionsspanish
11060 \renewcommand*{\glossaryname}{\textspanish{Glosario}}%
11061 \renewcommand*{\acronymname}{\textspanish{Siglas}}%
11062 \renewcommand*{\entryname}{\textspanish{Entrada}}%
11063 \renewcommand*{\descriptionname}{\textspanish{Descripci' on}}%
11064 \renewcommand*{\symbolname}{\textspanish{S'\i mbolo}}%
11065 \renewcommand*{\pagelistname}{\textspanish{Lista de p' aginas}}%
11066 \renewcommand*{\glssymbolsgroupname}{\textspanish{S'\i mbolos}}%

```

```

11067 \renewcommand*{\glnumbersgroupname}{\textspanish{N\'umeros}}%
11068 }%
11069 \edef\captionsspanish{\the\toks@}%
11070 }

```

French:

```

11071 \@ifundefined{captionsfrench}{}{%
11072 \expandafter\toks@\expandafter{\captionsfrench
11073 \renewcommand*{\glossaryname}{\textfrench{Glossaire}}%
11074 \renewcommand*{\acronymname}{\textfrench{Acronymes}}%
11075 \renewcommand*{\entryname}{\textfrench{Terme}}%
11076 \renewcommand*{\descriptionname}{\textfrench{Description}}%
11077 \renewcommand*{\symbolname}{\textfrench{Symbole}}%
11078 \renewcommand*{\pagelistname}{\textfrench{Pages}}%
11079 \renewcommand*{\glssymbolsgroupname}{\textfrench{Symboles}}%
11080 \renewcommand*{\glnumbersgroupname}{\textfrench{Nombres}}%
11081 }%
11082 \edef\captionsfrench{\the\toks@}%
11083 }

```

Danish:

```

11084 \@ifundefined{captionsdanish}{}{%
11085 \expandafter\toks@\expandafter{\captionsdanish
11086 \renewcommand*{\glossaryname}{\textdanish{Ordliste}}%
11087 \renewcommand*{\acronymname}{\textdanish{Akronymer}}%
11088 \renewcommand*{\entryname}{\textdanish{Symbolforklaring}}%
11089 \renewcommand*{\descriptionname}{\textdanish{Beskrivelse}}%
11090 \renewcommand*{\symbolname}{\textdanish{Symbol}}%
11091 \renewcommand*{\pagelistname}{\textdanish{Side}}%
11092 \renewcommand*{\glssymbolsgroupname}{\textdanish{Symboler}}%
11093 \renewcommand*{\glnumbersgroupname}{\textdanish{Tal}}%
11094 }%
11095 \edef\captionsdanish{\the\toks@}%
11096 }

```

Irish:

```

11097 \@ifundefined{captionsirish}{}{%
11098 \expandafter\toks@\expandafter{\captionsirish
11099 \renewcommand*{\glossaryname}{\textirish{Gluais}}%
11100 \renewcommand*{\acronymname}{\textirish{Acrainmneacha}}%
11101 \renewcommand*{\entryname}{\textirish{Ciall}}%
11102 \renewcommand*{\descriptionname}{\textirish{Tuaireasc}}%
11103 \renewcommand*{\symbolname}{\textirish{Comhartha}}%
11104 \renewcommand*{\glssymbolsgroupname}{\textirish{Comhartha\'{i}}}%
11105 \renewcommand*{\pagelistname}{\textirish{Leathanaigh}}%
11106 \renewcommand*{\glnumbersgroupname}{\textirish{Uimhreacha}}%
11107 }%
11108 \edef\captionsirish{\the\toks@}%
11109 }

```

Hungarian:

```

11110 \@ifundefined{captionsmagyar}{}{%

```

```

11111 \expandafter\toks@\expandafter{\captionsmagyar
11112 \renewcommand*{\glossaryname}{\textmagyar{Sz\'ojegyz\'ek}}%
11113 \renewcommand*{\acronymname}{\textmagyar{Bet\H uszavak}}%
11114 \renewcommand*{\entryname}{\textmagyar{Kifejez\'es}}%
11115 \renewcommand*{\descriptionname}{\textmagyar{Magyar\'azat}}%
11116 \renewcommand*{\symbolname}{\textmagyar{Jel\'ol\'es}}%
11117 \renewcommand*{\pagelistname}{\textmagyar{Oldalsz\'am}}%
11118 \renewcommand*{\glssymbolsgroupname}{\textmagyar{Jelek}}%
11119 \renewcommand*{\glsnumbersgroupname}{\textmagyar{Sz\'amjegyek}}%
11120 }%
11121 \edef\captionsmagyar{\the\toks@}%
11122 }

Polish
11123 \@ifundefined{captionspolish}{}{%
11124 \expandafter\toks@\expandafter{\captionspolish
11125 \renewcommand*{\glossaryname}{\textpolish{S\l ownnik termin\'ow}}%
11126 \renewcommand*{\acronymname}{\textpolish{Skr\'ot}}%
11127 \renewcommand*{\entryname}{\textpolish{Termin}}%
11128 \renewcommand*{\descriptionname}{\textpolish{Opis}}%
11129 \renewcommand*{\symbolname}{\textpolish{Symbol}}%
11130 \renewcommand*{\pagelistname}{\textpolish{Strony}}%
11131 \renewcommand*{\glssymbolsgroupname}{\textpolish{Symbole}}%
11132 \renewcommand*{\glsnumbersgroupname}{\textpolish{Liczby}}%
11133 }%
11134 \edef\captionspolish{\the\toks@}%
11135 }

Portugues
11136 \@ifundefined{captionsportuges}{}{%
11137 \expandafter\toks@\expandafter{\captionsportuges
11138 \renewcommand*{\glossaryname}{\textportuges{Gloss\'ario}}%
11139 \renewcommand*{\acronymname}{\textportuges{Siglas}}%
11140 \renewcommand*{\entryname}{\textportuges{Nota\c c\~ao}}%
11141 \renewcommand*{\descriptionname}{\textportuges{Descri\c c\~ao}}%
11142 \renewcommand*{\symbolname}{\textportuges{S\'imbolo}}%
11143 \renewcommand*{\pagelistname}{\textportuges{Lista de P\'aginas}}%
11144 \renewcommand*{\glssymbolsgroupname}{\textportuges{S\'imbolos}}%
11145 \renewcommand*{\glsnumbersgroupname}{\textportuges{N\'umeros}}%
11146 }%
11147 \edef\captionsportuges{\the\toks@}%
11148 }

```

7.3 Brazilian Dictionary

This is a dictionary file provided by Thiago de Melo for use with the package.

```
11149 \ProvidesDictionary{glossaries-dictionary}{Brazilian}
```

Provide Brazilian translations:

```

11150 \providetranslation{Glossary}{Gloss\'ario}
11151 \providetranslation{Acronyms}{Siglas}
11152 \providetranslation{Notation (glossaries)}{Nota\c c\~ao}

```

```

11153 \providetranslation{Description (glossaries)}{Descri\c c\~ao}
11154 \providetranslation{Symbol (glossaries)}{S\`imbolo}
11155 \providetranslation{Page List (glossaries)}{Lista de P\`aginas}
11156 \providetranslation{Symbols (glossaries)}{S\`imbolos}
11157 \providetranslation{Numbers (glossaries)}{N\`umeros}

```

7.4 Danish Dictionary

This is a dictionary file provided for use with the package.

```

11158 \ProvidesDictionary{glossaries-dictionary}{Danish}

```

Provide Danish translations:

```

11159 \providetranslation{Glossary}{Ordliste}
11160 \providetranslation{Acronyms}{Akronymer}
11161 \providetranslation{Notation (glossaries)}{Symbolforklaring}
11162 \providetranslation{Description (glossaries)}{Beskrivelse}
11163 \providetranslation{Symbol (glossaries)}{Symbol}
11164 \providetranslation{Page List (glossaries)}{Side}
11165 \providetranslation{Symbols (glossaries)}{Symboler}
11166 \providetranslation{Numbers (glossaries)}{Tal}

```

7.5 Dutch Dictionary

This is a dictionary file provided for use with the package.

```

11167 \ProvidesDictionary{glossaries-dictionary}{Dutch}

```

Provide Dutch translations:

```

11168 \providetranslation{Glossary}{Woordenlijst}
11169 \providetranslation{Acronyms}{Acroniemen}
11170 \providetranslation{Notation (glossaries)}{Benaming}
11171 \providetranslation{Description (glossaries)}{Beschrijving}
11172 \providetranslation{Symbol (glossaries)}{Symbool}
11173 \providetranslation{Page List (glossaries)}{Pagina's}
11174 \providetranslation{Symbols (glossaries)}{Symbolen}
11175 \providetranslation{Numbers (glossaries)}{Cijfers}

```

7.6 English Dictionary

This is a dictionary file provided for use with the package.

```

11176 \ProvidesDictionary{glossaries-dictionary}{English}

```

Provide English translations:

```

11177 \providetranslation{Glossary}{Glossary}
11178 \providetranslation{Acronyms}{Acronyms}
11179 \providetranslation{Notation (glossaries)}{Notation}
11180 \providetranslation{Description (glossaries)}{Description}
11181 \providetranslation{Symbol (glossaries)}{Symbol}
11182 \providetranslation{Page List (glossaries)}{Page List}
11183 \providetranslation{Symbols (glossaries)}{Symbols}
11184 \providetranslation{Numbers (glossaries)}{Numbers}

```

7.7 French Dictionary

This is a dictionary file provided for use with the package.

```
11185 \ProvidesDictionary{glossaries-dictionary}{French}
```

Provide French translations:

```
11186 \providetranslation{Glossary}{Glossaire}
11187 \providetranslation{Acronyms}{Acronymes}
11188 \providetranslation{Notation (glossaries)}{Terme}
11189 \providetranslation{Description (glossaries)}{Description}
11190 \providetranslation{Symbol (glossaries)}{Symbole}
11191 \providetranslation{Page List (glossaries)}{Pages}
11192 \providetranslation{Symbols (glossaries)}{Symboles}
11193 \providetranslation{Numbers (glossaries)}{Nombres}
```

7.8 German Dictionary

This is a dictionary file provided for use with the package.

```
11194 \ProvidesDictionary{glossaries-dictionary}{German}
```

Provide German translations (quite a few variations were suggested for German; I settled on the following):

```
11195 \providetranslation{Glossary}{Glossar}
11196 \providetranslation{Acronyms}{Akronyme}
11197 \providetranslation{Notation (glossaries)}{Bezeichnung}
11198 \providetranslation{Description (glossaries)}{Beschreibung}
11199 \providetranslation{Symbol (glossaries)}{Symbol}
11200 \providetranslation{Page List (glossaries)}{Seiten}
11201 \providetranslation{Symbols (glossaries)}{Symbole}
11202 \providetranslation{Numbers (glossaries)}{Zahlen}
```

7.9 Irish Dictionary

This is a dictionary file provided for use with the package.

```
11203 \ProvidesDictionary{glossaries-dictionary}{Irish}
```

Provide Irish translations:

```
11204 \providetranslation{Glossary}{Gluais}
11205 \providetranslation{Acronyms}{Acrainmneacha}
11206 \providetranslation{Notation (glossaries)}{Ciall}
11207 \providetranslation{Description (glossaries)}{Tuairisc}
11208 \providetranslation{Symbol (glossaries)}{Comhartha}
11209 \providetranslation{Page List (glossaries)}{Leathanaigh}
11210 \providetranslation{Symbols (glossaries)}{Comhartha'\{i}}
11211 \providetranslation{Numbers (glossaries)}{Uimhreacha}
```

7.10 Italian Dictionary

This is a dictionary file provided for use with the package.

```
11212 \ProvidesDictionary{glossaries-dictionary}{Italian}
```

Provide Italian translations:

```
11213 \providetranslation{Glossary}{Glossario}
11214 \providetranslation{Acronyms}{Acronimi}
11215 \providetranslation{Notation (glossaries)}{Nomenclatura}
11216 \providetranslation{Description (glossaries)}{Descrizione}
11217 \providetranslation{Symbol (glossaries)}{Simbolo}
11218 \providetranslation{Page List (glossaries)}{Elenco delle pagine}
11219 \providetranslation{Symbols (glossaries)}{Simboli}
11220 \providetranslation{Numbers (glossaries)}{Numeri}
```

7.11 Magyar Dictionary

This is a dictionary file provided for use with the package.

```
11221 \ProvidesDictionary{glossaries-dictionary}{Magyar}
```

Provide translations:

```
11222 \providetranslation{Glossary}{Sz\'ojegyz\'ek}
11223 \providetranslation{Acronyms}{Bet\H uszavak}
11224 \providetranslation{Notation (glossaries)}{Kifejez\'es}
11225 \providetranslation{Description (glossaries)}{Magyar\'azat}
11226 \providetranslation{Symbol (glossaries)}{Jel\'ol\'es}
11227 \providetranslation{Page List (glossaries)}{Oldalsz\'am}
11228 \providetranslation{Symbols (glossaries)}{Jelek}
11229 \providetranslation{Numbers (glossaries)}{Sz\'amjegyek}
```

7.12 Polish Dictionary

This is a dictionary file provided for use with the package.

```
11230 \ProvidesDictionary{glossaries-dictionary}{Polish}
```

Provide Polish translations:

```
11231 \providetranslation{Glossary}{S{\l}ownik termin\'ow}
11232 \providetranslation{Acronyms}{Skr\'ot}
11233 \providetranslation{Notation (glossaries)}{Termin}
11234 \providetranslation{Description (glossaries)}{Opis}
11235 \providetranslation{Symbol (glossaries)}{Symbol}
11236 \providetranslation{Page List (glossaries)}{Strony}
11237 \providetranslation{Symbols (glossaries)}{Symbole}
11238 \providetranslation{Numbers (glossaries)}{Liczby}
```

7.13 Serbian Dictionary

This dictionary was provided by Zoran Filipovic.

```
11239 \ProvidesDictionary{glossaries-dictionary}{Serbian}
11240 \providetranslation{Glossary}{Mali re\vnik}
11241 \providetranslation{Acronyms}{Skra\'cenice}
11242 \providetranslation{Notation (glossaries)}{Oznaka}
11243 \providetranslation{Description (glossaries)}{Opis}
11244 \providetranslation{Symbol (glossaries)}{Simbol}
```

```

11245 \providetranslation{Page List (glossaries)}{Stranica}
11246 \providetranslation{Symbols (glossaries)}{Simboli}
11247 \providetranslation{Numbers (glossaries)}{Brojevi}

```

7.14 Spanish Dictionary

This is a dictionary file provided for use with the package.

```
11248 \ProvidesDictionary{glossaries-dictionary}{Spanish}
```

Provide Spanish translations:

```

11249 \providetranslation{Glossary}{Glosario}
11250 \providetranslation{Acronyms}{Siglas}
11251 \providetranslation{Notation (glossaries)}{Entrada}
11252 \providetranslation{Description (glossaries)}{Descripci'on}
11253 \providetranslation{Symbol (glossaries)}{S'\i mbolo}
11254 \providetranslation{Page List (glossaries)}{Lista de p'aginas}
11255 \providetranslation{Symbols (glossaries)}{S'\i mbolos}
11256 \providetranslation{Numbers (glossaries)}{N'umeros}

```

Glossary

`makeindex` An indexing application. [10](#), [23](#)

`xindy` An flexible indexing application with multilingual support written in Perl. [10](#), [23](#)

Change History

1.01	General: Added range facility in format key 92	Changed the default value of the sort key to just the value of the name key 69
	<code>\writeist</code> : Added spaces after <code>\delimN</code> and <code>\delimR</code> in ist file 147	<code>\glsmakefirstuc</code> : new 245
1.03	<code>\makefirstuc</code> : changed 'pro- tected@edef to 'def 244	1.06 General: now requires etoolbox . 244
1.04	General: Added <code>\glstextformat</code> 78	<code>\capitalisewords</code> : new 245
1.05	<code>\glossarysection</code> : added <code>\@mkboth to \glossarysection</code> 35	<code>\xcapitalisewords</code> : new 246
	<code>\gls@defglossaryentry</code> :	1.07 <code>\@gls@link</code> : fixed bug caused by <code>\theglentrycounter</code> set- ting the page number too soon 90
		<code>\glsadd</code> : fixed bug caused by <code>\theglentrycounter</code> set- ting the page number too soon 145

1.08	General: Added babel support ... 30	descriptionplural:new 56
	\capitalisewords: made robust	\gls@defglossaryentry:
 245	Changed default first plural to
	listgroup: changed listgroup	be first key with s appended
	style to use \glsgetgrouptitle	(was text key with s appended) 69
 251	descriptionplural support
	altlistgroup: changed al-	added 68
	tlistgroup style to use	symbolplural support added .. 69
	\glsgetgrouptitle 252	\Glsentrydescplural: New .. 139
	\makefirstuc: made robust ... 244	\glsentrydescplural: New .. 139
1.1		\Glsentrysymbolplural: New 140
	\@glossarysection: numbered	\glsentrysymbolplural: New 140
	sections and auto label added 36	\SetDescriptionFootnoteAcronymStyle:
	\@gls@tmpb: changed \toksdef	Added \protect before
	to \newtoks 95	\footnote and \glslink . 217
	\@gls@toc: numberline added .. 38	\SetFootnoteAcronymStyle:
	\@p@glossarysection: num-	Added \protect before
	bered sections and auto label	\footnote and \glslink . 224
	added 37	symbolplural: new 57
	1.13	
	General: Added support for trans-	General: Add Polish support 352, 355
	lator package 30	fixed bug that ignored 3rd pa-
	amsgen now loaded (\new@ifnextchar	rameter 111–121
	needed) 4	\ACRfullpl: new 199
	translate: translate option	\Acrfullpl: new 199
	added 21	\acrfullpl: new 198
	\setglossarysection: new ... 36	\acrpluralsuffix: New 196
	numberedsection: numbered-	\gls@defglossaryentry:
	section package option added . 6	Changed default first value .. 69
	numberline: numberline option	Changed default firstplural
	added 5	value 69
1.12		Removed restriction on only
	\@GLSpl: now uses \glsentrydescplural	using \newglossaryentry in
	and \glsentrysymbolplural	the preamble 73
	instead of \glsentrydesc	\newacronym: Removed re-
	and \glsentrysymbol 109	striction on only using
	\@GLspl@: now uses \glsentrydescplural	\newacronym in the preamble 196
	and \glsentrysymbolplural	1.14
	instead of \glsentrydesc	\@gls@hypergroup: new 247
	and \glsentrysymbol 108	General: added nonnumberlist key
	\@GLspl@: now uses \glsentrydescplural	to \printglossary 182
	and \glsentrysymbolplural	added numberedsection key to
	instead of \glsentrydesc	\printglossary 181
	and \glsentrysymbol 106	\firstacronymfont: new 200
	General: added check for	\glsautoprefix: new 6
	\hypertarget separate to	\glsnavhyperlink: changed
	\hyperlink (memoir de-	'edef to 'protected@edef ... 246
	finies \hyperlink but not	\glsnavhypertarget: added
	\hypertarget) 100	write to aux file 247

<code>\glsnavigation:</code> changed to only use labels for groups that are present	248	scroll off the top of the page	100
1.15		<code>\gls@defglossaryentry:</code> Changed def to let	69
<code>\@gls@link:</code> added <code>\glslabel</code> .	90	1.17	
General: Added <code>\glssettoctitle</code>	30	<code>\@do@wrglossary:</code> new	163
<code>\gls@defglossaryentry:</code> check for <code>\@glo@first</code> in description	72	<code>\@do@seeglossary:</code> new	165
check for <code>\@glo@text</code> in symbol	72	<code>\@glo@storeentry:</code> new	74
<code>\gls@hypergroupprerun:</code> new .	247	<code>\@glossary:</code> changed definition to use <code>\index</code> instead of <code>\@index</code>	161
<code>\glsnavhypertarget:</code> added check if rerun required	247	<code>\@glsdefaultplural:</code> new	60
<code>\glssettoctitle:</code> new	29	<code>\@glsdefaultsort:</code> new	60
<code>\printglossary:</code> changed the way the TOC title is set	168	<code>\@glshypernumber:</code> new	193
1.16		<code>\@glsnoname:</code> new	59
<code>\@GLS@:</code> Test glossary type is <code>\acronymtype</code> in addition to checking if footnote option has been used	105	<code>\@glsnonextpages:</code> new	182
<code>\@GLSpl:</code> Test glossary type is <code>\acronymtype</code> in addition to checking if footnote option has been used	109	<code>\@wrglossary:</code> modified to allow for xindy support	162
<code>\@Gls@:</code> Test glossary type is <code>\acronymtype</code> in addition to checking if footnote option has been used	104	General: added Brazilian dictionary	355
<code>\@Glspl@:</code> Test glossary type is <code>\acronymtype</code> in addition to checking if footnote option has been used	108	Added Brazilian support	352
<code>\@gls@:</code> Test glossary type is <code>\acronymtype</code> in addition to checking if footnote option has been used	102	added xindy support	23
<code>\@glsdisp:</code> Test glossary type is <code>\acronymtype</code> in addition to checking if footnote option has been used	110	parent: new	58
<code>\@glspl@:</code> Test glossary type is <code>\acronymtype</code> in addition to checking if footnote option has been used	106	see: new	58
<code>\@glstarget:</code> raised the hypertarget so the target text doesn't		<code>\gls@defglossaryentry:</code> added nonumberlist key	69
		added parent key	69
		added see key	69
		Stored main part of entry format when entry is defined	73
		<code>\gls@suffixF:</code> new	33
		<code>\gls@suffixFF:</code> new	33
		<code>\glshyperlink:</code> new	145
		<code>\glshypernumber:</code> modified to allow material to be attached to location	192
		<code>\glsnavhyperlink:</code> replaced 'hyperlink' to '@glslink'	246
		<code>\glsnavhypertarget:</code> replaced 'hypertarget' to '@glstarget' .	247
		<code>\glssee:</code> new	166
		<code>\glsseeformat:</code> new	166
		<code>\glsSetSuffixF:</code> new	33
		<code>\glsSetSuffixFF:</code> new	34
		<code>\ifglsxindy:</code> new	23
		<code>\istfilename:</code> added xindy support	32
		<code>\newglossarystyle:</code> made <code>\newglossarystyle long</code> .	191
		<code>\nopostdesc:</code> new	31

nonumberlist: new	58	2.01	
\printglossary: added check to determine if \printglossary is already defined	168		\@gls@link: moved \@do@wrglossary before term is displayed to pre- vent unwanted whatsit
added print language to aux file	168		91
order: order package option			\forall glossaries: replaced \ifthenelse with \ifx
added	23		47
\writeist: added xindy support	147		\forall glossentries: replaced \ifthenelse with \ifx
1.18			47
\@gls@loadlist: new	8		\glsdefmain: new
\@gls@loadlong: new	8		12
\@gls@loadsuper: new	8		\glsdescwidth: changed \linewidth to \hspace . 254, 269
\@gls@loadtree: new	8		\glslistdottedwidth: changed \linewidth to \hspace
\gls@defglossaryentry:			253
Changed default value of sort			\glslistwidth: changed \linewidth to \hspace . 254, 269
to \@glsdefaultsort	69		nomain: added nomain package option
moved sort sanitization to			13
\newglossaryentry	72		\writeist: removed item_02 - no such makeindex key
\glstarget: new	186		152
\oldacronym: new	195	2.02	
nolist: new	8		\@printglossary: suppressed warning globally rather than locally
nolong: new	8		170
sort: moved sanitization to			General: Changed Brazil to Brazil- ian
\newglossaryentry	56		355
nostyles: new	8		false will prevent automatic loading of translator package
nosuper: new	8		27
notree: new	8		\glossarysection: changed \@mkboth to \glossarymark
1.19			35
\gls@clearpage: new	38		\gls@glossarymark: New
\gls@disp: new	110	2.03	
\SetDescriptionAcronymStyle:			\@GLS@: Added check for hyper- first
changed \acronymfont to			105
use \textsmaller instead of			\@GLS@: Added check for hyper- first
\smaller	222		109
\SetDescriptionFootnoteAcronymStyle:			\@GLS@: Added check for hyper- first
changed \acronymfont to			104
use \textsmaller instead of			\@GLS@: Added check for hyper- first
\smaller	217		108
\SetFootnoteAcronymStyle:			\@GLS@: Added check for hyper- first
changed \acronymfont to			102
use \textsmaller instead of			\@gls@link: new
\smaller	224		90
\SetSmallAcronymStyle:			\@gls@link: added \leavevmode
changed \acronymfont to			90
use \textsmaller instead of			Moved entry existence check to avoid duplicate code
\smaller	227		90
1.2			\@gls@disp: Added check for hy- perfirst
General: fixed bug in ngerman			110
captions	349		

\@glsp1@: Added check for hyper-first	106	\glentryuseriv: new	142
\glsglossarymark: Added check to see if it's already defined ..	35	\Glsentryuseriv: new	142
hyperfirst: new	22	\glentryuseriv: new	142
2.04		\glentryuseriv: new	142
\@GLS@: Changed test to check if glossary type has been identified as a list of acronyms ...	105	\newglossary: added check to determine if \gls@<type>@display and \gls@<type>@displayfirst have been defined.	54
\@GLSp1: Changed test to check if glossary type has been identified as a list of acronyms ...	109	\SetAcronymLists: new	15
\@GLs@: Changed test to check if glossary type has been identified as a list of acronyms ...	104	\SetDefaultAcronymDisplayStyle: new	214
\@GLsp1@: Changed test to check if glossary type has been identified as a list of acronyms ..	108	\SetDefaultAcronymStyle: new	215
\@glossaryentryfield: new ..	74	\SetDescriptionAcronymDisplayStyle: new	219
\@glossarysubentryfield: new	74	\SetDescriptionDUAAcronymDisplayStyle: new	218
\@gl@: Changed test to check if glossary type has been identified as a list of acronyms ...	102	\SetDescriptionFootnoteAcronymDisplayStyle: new	216
\@gl@acronymlists: new	14	\SetDUADisplayStyle: new ..	227
\@gl@disp: Changed test to check if glossary type has been identified as a list of acronyms ..	110	\SetFootnoteAcronymDisplayStyle: new	222
\@glsp1@: Changed test to check if glossary type has been identified as a list of acronyms ..	106	\SetSmallAcronymDisplayStyle: new	224
\@newglossaryentryposthook: new	73	2.05	
\@newglossaryentryprehook: new	73	\@gl@disp: Added closing brace. Patch provided by Sergiu Dotenco	110
acronymlists: new	15	Removed spurious brace. Patch provided by Sergiu Dotenco ..	111
\DeclareAcronymList: new ...	14	\writeist: Added \string before opening and closing braces. Patch provided by Segiu Dotenco	152
\DefineAcronymSynonyms: new	212	2.06	
\gls@defglossaryentry: added user1-6 keys	69	\altnewglossary: new	54
\glsadd: fixed bug that ignored counter	145	\CustomAcronymFields: new ..	230
\Glsentryuseri: new	141	\CustomNewAcronymDef: new ..	230
\Glsentryuseri: new	141	\SetCustomDisplayStyle: new	229
\Glsentryuserii: new	141	\SetCustomStyle: new	230
\Glsentryuserii: new	141	2.07	
\Glsentryuseriii: new	142	General: glssadd format key stored in \@glsnumberformat (was mistakenly stored in \@glo@format)	145
\Glsentryuseriii: new	141	3.0	
\Glsentryuseriv: new	142	\@do@wrglossary: added check for hyper location prefix ...	164

modified to use new format ..	163	\acrfullpl: added starred ver-	198
\@@glossarysec: replaced		sion	198
\@ifundefined with		\acrlinkfootnote:new	215
\ifcsundef	5	\acrnolinkfootnote:new ...	215
\@do@seeglossary: Sanitize and		\addglossarytocaptions: re-	
escape cross-referencing in-		placed \@ifundefined with	
formation	165	\ifcsundef	30
\@gls@counterwithin:new	9	savewrites:new	24
\@gls@ifinlist:new	39	see:added \@glo@seeautonumberlist	
\@gls@link:added \@gls@saveentrycounter.....	58	seeautonumberlist:new	7
.....	91	\glossarysection: replaced	
added \@gls@setsort	91	\@ifundefined with	
\@gls@saveentrycounter:new	91	\ifcsundef	35
\@gls@setupsort@def:new ...	11	\glossarystyle: replaced	
\@gls@setupsort@standard:		\@ifundefined with	
new	10	\ifcsundef	191
\@gls@setupsort@use:new ...	11	\gls@codepage: replaced	
\@gls@xdy@locationlist:new	42	\@ifundefined with	
\@glslink:replaced \@ifundefined		\ifcsundef	24
with \ifcsundef	100	\gls@defglossaryentry: added	
\@glsnextpages:new	183	\@gls@defsort	72
\@makeglossary: Added check		added short and long keys	69
for savewrites	154	replaced \@ifundefined with	
\@print@glossary: replaced		\ifcsundef	70
\@ifundefined with		\gls@doclearpage: replaced	
\ifcsundef	171	\@ifundefined with	
\@printglossary: added		\ifcsundef	37
\currentglossary	169	\glsadd:added \@gls@saveentrycounter	
added \glsnextpages	169	146
make toctitle default to title ..	169	\GlsAddXdyCounters:new	39
\@set@glo@numformat: added		\glentrycounterlabel:new	185
4th argument	92	\glentryitem:new	185
\@wrglossary: modified to take		\Glentrylong:new	143
into account savewrites	162	\glentrylong:new	143
\@xdyattributelist:new	39	\Glentrylongpl:new	143
General: added prefix to hyperlink		\glentrylongpl:new	143
.....	194	\Glentryshort:new	142
etoolbox now loaded	4	\glentryshort:new	142
replaced \@ifundefined with		\Glentryshortpl:new	143
\ifcsundef	28, 89, 181	\glentryshortpl:new	143
\acrfootnote:new	215	\glsgetgrouptitle: re-	
\ACRfull: added starred version	198	placed \@ifundefined with	
\Acrfull: added starred version	197	\ifcsundef	189
\acrfull: added starred version	197	\gls glossarymark: replaced	
\ACRfullpl: added starred ver-		\@ifundefined with	
sion	199	\ifcsundef	35
\Acrfullpl: added starred ver-		\glshyperlink: changed de-	
sion	199	fault from \glentryname to	

<code>\glstrytext</code>	145	<code>long:new</code>	59
<code>\glshypernumber:</code>	replaced	<code>longplural:new</code>	59
<code>\@ifundefined</code>	with	<code>nonumberlist:now boolean</code> ...	58
<code>\ifcsundef</code>	192	<code>sort:new</code>	10
<code>\glstrnumberformat:</code>	replaced	<code>counter:replaced \@ifundefined</code>	
<code>\@ifundefined</code>	with	<code>with \ifcsundef</code>	57
<code>\ifcsundef</code>	34	<code>\printglossary:</code>	replaced
<code>\glstrsrefentry:new</code>	184	<code>\@ifundefined</code>	with
<code>\glstrsresetsubentrycounter:</code>		<code>\ifcsundef</code>	168
<code>new</code>	184	<code>\SetDescriptionFootnoteAcronymDisplayStyle:</code>	
<code>\glssseeitem:</code>	hyperlink uses	<code>expanded options link op-</code>	
<code>\glssseeitemformat</code>	instead	<code>tions</code>	216
<code>of \glstryname</code>	167	<code>\setentrycounter:</code>	added op-
<code>\glssseeitemformat:new</code>	167	<code>tional argument</code>	190
<code>\glsssortnumberfmt:new</code>	10	<code>\showacronymlists:new</code>	236
<code>\glssstepentry:new</code>	184	<code>\showglocounter:new</code>	233
<code>\glssstepsubentry:new</code>	184	<code>\showgloDESC:new</code>	234
<code>\glsssubentrycounterlabel:</code>		<code>\showgloDESCplural:new</code> ...	234
<code>new</code>	185	<code>\showglofirst:new</code>	232
<code>\glsssubentryitem:new</code>	185	<code>\showglofirstpl:new</code>	232
<code>theglossary:replaced \@ifundefined</code>		<code>\showgloflag:new</code>	235
<code>with \ifcsundef</code>	185	<code>\showgloindex:new</code>	235
<code>short:new</code>	59	<code>\showglolevel:new</code>	232
<code>shortplural:new</code>	59	<code>\showgloNAME:new</code>	234
<code>\ifglossaryexists:</code>	re-	<code>\showgloparent:new</code>	231
<code>placed \@ifundefined with</code>		<code>\showgloplural:new</code>	232
<code>\ifcsundef</code>	48	<code>\showglosort:new</code>	234
<code>\ifglstryexists:</code>	re-	<code>\showglossaries:new</code>	236
<code>placed \@ifundefined with</code>		<code>\showglossarycounter:new</code> .	236
<code>\ifcsundef</code>	49	<code>\showglossaryentries:new</code> .	237
<code>\istfile: deprecated</code>	160	<code>\showglossaryin:new</code>	236
<code>glossaryentry:new</code>	183	<code>\showglossaryout:new</code>	236
<code>glossarysubentry:new</code>	183	<code>\showglossarytitle:new</code> ...	236
<code>\newglossary:added \@glstrsortcount</code>		<code>\showglosymbol:new</code>	234
<code>.....</code>	54	<code>\showglosymbolplural:new</code> .	235
<code>replaced \@ifundefined with</code>		<code>\showgloTEXT:new</code>	232
<code>\ifcsundef</code>	54	<code>\showgloTYPE:new</code>	232
<code>\newglossaryentry:</code>	replaced	<code>\showgloUSERi:new</code>	233
<code>\DeclareRobustCommand</code>		<code>\showgloUSERii:new</code>	233
<code>with \newrobustcmd</code>	62	<code>\showgloUSERiii:new</code>	233
<code>\newglossarystyle:</code>	re-	<code>\showgloUSERiv:new</code>	233
<code>placed \@ifundefined with</code>		<code>\showgloUSERv:new</code>	233
<code>\ifcsundef</code>	191	<code>\showgloUSERvi:new</code>	234
<code>entrycounter:new</code>	9	<code>subentrycounter:new</code>	9
<code>entrycounterwithin:new</code>	9	<code>\writeist:</code>	added xindy-only
<code>\oldacronym:replaced \@ifundefined</code>		<code>macro definitions to glossary</code>	
<code>with \ifcsundef</code>	195	<code>open tag</code>	150
<code>compatible-2.07: compatible-</code>		<code>modified to support new for-</code>	
<code>2.07 option added</code>	25	<code>mat</code>	147

3.01	
\@glswritefiles: added check for empty glossaries	160
General: made robust	104
\ACRfull: made robust	198
\Acrfull: made robust	197
\acrfull: made robust	196
\acrfullformat: removed \acronymfont as it should al- ready be set in the second ar- gument.	197
\ACRfullpl: made robust	199
\Acrfullpl: made robust	199
\acrfullpl: made robust	198
\ACRlong: made robust	134
\Acrlong: made robust	133
\acrlong: made robust	132
\ACRlongpl: made robust	137
\Acrlongpl: made robust	136
\acrlongpl: made robust	135
\ACRshort: made robust	129
\Acrshort: made robust	128
\acrshort: made robust	127
\ACRshortpl: made robust	132
\Acrshortpl: made robust	131
\acrshortpl: made robust	130
\Gls: made robust	103
\glsadd: made robust	145
\glsaddall: made robust	146
\GLSdesc: made robust	117
\Glsdesc: made robust	117
\glsdesc: made robust	117
\GLSdescplural: made robust .	118
\Glsdescplural: made robust .	118
\glsdescplural: made robust .	118
\glsfirst: made robust	112
\GLSfirstplural: made robust	115
\Glsfirstplural: made robust	115
\glsfirstplural: made robust	114
\glslink: made robust	90
\GLSname: made robust	116
\Glsname: made robust	116
\glsname: made robust	116
\GLSpl: made robust	108
\Glspl: made robust	107
\glspl: made robust	106
\GLSplural: made robust	114
\GLSsymbol: made robust	119
\Glsymbol: made robust	119
\glsymbol: made robust	119
\GLSsymbolplural: made robust	121
\Glsymbolplural: made robust	120
\glsymbolplural: made robust	120
\Glstext: made robust	112
\glstext: made robust	111
\GLSuseri: made robust	122
\Glsuseri: made robust	121
\glsuseri: made robust	121
\GLSuserii: made robust	123
\Glsuserii: made robust	122
\glsuserii: made robust	122
\GLSuseriii: made robust	124
\Glsuseriii: made robust	123
\glsuseriii: made robust	123
\GLSuseriv: made robust	125
\Glsuseriv: made robust	124
\glsuseriv: made robust	124
\GLSuserv: made robust	126
\Glsuserv: made robust	126
\glsuserv: made robust	125
\GLSuservi: made robust	127
\Glsuservi: made robust	127
\glsuservi: made robust	126
3.02	
\@do@wrglossary: changed \@glslocref to \theglscopycounter	164
\@do@wrglossary: changed \@do@wr@glossary to test for indexonlyfirst option; put old \@do@wr@glossary code into \@do@wrglossary	162
\@gls@missingnumberlist: new	60
\@glswritefiles: added check for existence of token in case \makeglossaries has been omitted	160
\@printglossary: add a way to fetch current entry label ...	170
\@wrglossary: added check for glossary file defined	162
General: added check for polyglos- sia	27
reversed order of package check	31

savenumberlist:new	7	altlongragged4col: added	
ucmark:new	9	check for glsnogroupskip ..	264
\gls@defglossaryentry: added		altsuperragged4col: added	
numberlist element	72	check for glsnogroupskip ..	281
\gls@save@numberlist:new ..	167	alttree: added check for	
\glsdisplaynumberlist:new ..	144	glsnogroupskip	289
\glsentrycounter: set default		index: added check for	
value	91	glsnogroupskip	283
\Glsentryfull: fixed bug (re-		nogroupskip:new	9
placed \glsentryshortpl		long: added check for	
with \glsentryshort)	143	glsnogroupskip	255
\glsentryfullpl: fixed bug (re-		long3col: added check for	
placed \glsentryshort with		glsnogroupskip	256
\glsentryshortpl)	143	long4col: added check for	
\glsentrynumberlist:new ..	144	glsnogroupskip	258
\glsmoveentry:new	73	longragged: added check for	
\glsnumlistlastsep:new ...	145	glsnogroupskip	261
\glsnumlistsep:new	145	longragged3col: added check	
\glsresetsubentrycounter:		for glsnogroupskip	262
new	184	no postdot:new	9
\ifglshaschildren:new	50	tree: added check for	
\ifglshasparent:new	50	glsnogroupskip	284
\makeglossaries: added list		treenoname: added check for	
parser	155	glsnogroupskip	286
indexonlyfirst:new	22	super: added check for	
\renewglossarystyle:new ..	192	glsnogroupskip	270
\showglossaryentries: fixed		super3col: added check for	
misspelt command	237	glsnogroupskip	272
\SmallNewAcronymDef: fixed		super4col: added check for	
broken short and long plural	225	glsnogroupskip	273
3.03		superragged: added check for	
\@gls@sanitizesort:new	18	glsnogroupskip	277
\@gls@setupsort@standard:		superragged3col: added check	
used \@gls@sanitizesort ..	10	for glsnogroupskip	279
\@printglossary: allow title to		3.04	
override default toctitle	169	\@do@wrglossary: changed	
General: allow title to set toctitle	181	\theglsentrycounter back	
\glsinlinedescformat:new ..	250	to \@glslocref	164
\glsinlineemptydescformat:		\@do@wrglossary: modified to	
new	250	compensate for possible incor-	
\glsinlinenameformat:new ..	250	rect page number	163
\glsinlinepostchild:new ..	250	\@gls@escbsdq: unsani-	
\glsinlinesubdescformat:		tize \gls@numberpage,	
new	250	\gls@alphpage, \gls@Alphpage	
\glsinlinesubnameformat:		and \gls@romanpage	94
new	250	\@print@glossary: Moved aux	
\glspostinline: replaced “.”		write to end of document to	
with \glspostdescription	250	prevent unwanted whatsit oc-	
		curing here.	171

General: Added check for doc	3.07	
package	4	\@gls@link: fixed bug that failed to find entry in list
added datatool-base as a required package	4	
added local key	89	\glossarypreamble: modified to work with \setglossarypreamble
\gls@Alphpage: new	163	
\gls@alphpage: new	163	\gls@docclearpage: added check for openright
\gls@disablepagerefexpansion: new	162	\glspostdescription: Added spacefactor code
\gls@numberpage: new	163	
\gls@protected@pagefmts: new	162	\GlsSetXdyCodePage: Added check for fontspec
\gls@romanpage: new	163	\SetDescriptionAcronymDisplayStyle: now using \glsdoparenifnotempty
\glsdefmain: added check for doc package	12	
\glsorg@endtheglossary: new	5	\setglossarypreamble: new
\glsorg@glossary: new	4	3.08a
\glsorg@theglossary: new	5	\@glo@storeentry: no longer need to check for special characters in any of the fields other than sort
\glsorg@wrglossary: new	4	updated for \glossentry
altlist: replaced \newline with paragraph break	252	\@glossaryentryfield: switched to \glossentry
\PrintChanges: new	5	\@glossarysubentryfield: switched to \subglossentry
3.05		General: added nogroupskip key to \printglossary
\@do@wrglossary: add Roman case. Fixed bugs in the else statements	163	removed definition of \@glossaryentryfield
\@gls@link: added check for "no-hypertypes"	91	removed definition of \@glossarysubentryfield
\@gls@nohyperlist: new	16	\compatibleglossentry: new
mcolalttree: replaced '2' with \glsmcols	268	\compatiblesubglossentry: new
mcolindex: replaced '2' with \glsmcols	266	\glossaryentryfield: deprecated
mcoltree: replaced '2' with \glsmcols	266	\Glossentrydesc: new
mcoltreename: replaced '2' with \glsmcols	267	\glossentrydesc: new
\gls@protected@pagefmts: added Roman to list	162	\Glossentryname: new
\gls@Romanpage: new	163	\glossentryname: new
\GlsDeclareNoHyperList: new	16	\Glossentrysymbol: new
\glsgetgrouplabel: fixed bug (typo in \equal)	190	\glossentrysymbol: new
\nopostdesc: made robust	31	\gls@assign@desc@field: new
nohypertypes: new	16	\gls@assign@descplural@field: new
3.06		\gls@assign@field: new
\@xdy@main@language: Changed back to using \language	24	\gls@ifnotmeasuring: new
\findrootlanguage: Obsoleted	45	\glsaddallunused: new

\glsexpandfields: new	62	3.09a	
\glsnoexpandfields: new	62		\@gls@assign@symbolplural@field:
\glssee: made robust	166		new
\glsseeformat: made robust ..	166		18
\glsseeitem: made robust	167		\@gls@default@value: new ...
\glsseelist: made robust	166		57
\ifglsdescsuppressed: new ..	50		\Glsentrydesc: made robust ..
\ifglshasdesc: new	50		139
\ifglshassymbol: new	51		\Glsentrydescplural: made ro-
list: updated list style to			bust
use \glossentry and			139
\subglossentry	251		\Glsentryfirst: made robust .
listdotted: updated listdotted			140
style to use \glossentry and			\Glsentryfirstplural: made
\subglossentry	253		robust
altlist: updated altlist style			141
to use \glossentry and			\Glsentryfull: made robust ..
\subglossentry	252		143
altlongragged4col: updated			\Glsentryfullpl: made robust
to use \glossentry and			143
\subglossentry	264		\Glsentrylong: made robust ..
alttree: updated to use			143
\glossentry and \subglossentry			\Glsentrylongpl: made robust
.....	287		143
index: added paragraph break at			\Glsentryname: made robust ..
end of environment	282		139
updated to use \glossentry			\Glsentryplural: made robust
and \subglossentry	282		140
inline: updated inline style			\Glsentryshort: made robust .
to use \glossentry and			142
\subglossentry	249		\Glsentryshortpl: made robust
long: updated to use \glossentry		
and \subglossentry	254		143
longragged: updated to			\Glsentrysymbol: made robust
use \glossentry and			140
\subglossentry	261		\Glsentrysymbolplural: made
longragged3col: updated			robust
to use \glossentry and			140
\subglossentry	262		\Glsentrytext: made robust ..
tree: updated to use \glossentry			139
and \subglossentry	284		\Glsentryuseri: made robust .
\setglossarystyle: new	191		141
\setglossentrycompatibility:			\Glsentryuserii: made robust
new	188		141
superragged: updated to			\Glsentryuseriii: made robust
use \glossentry and		
\subglossentry	277		142
		3.10a	\Glsentryuseriv: made robust
			142
			\Glsentryuserv: made robust .
			142
			\Glsentryuservi: made robust
			142
			\glstextup: new
			196
			\if@gls@docloaded: Add a fix for
			\RecordChanges
			4
			\ifglshassymbol: changed test
			to check for \@gls@default@symbol
		
			51
			\@gls@keymap: new
			64
			\@gls@provide@newglossary:
			new
			53
			\@gls@writedef: new
			63
			\@gls@defaultplural: Obsolete .
			60
			\@gls@nodesc: new
			60
			\@print@glossary: Added
			providecommand code to aux
			file
			171
			\gls@assign@type@field: new
			17

<code>\gls@defglossaryentry:</code>	change to using <code>\glsentryfmt</code>
Changed to using <code>\@gls@default@value</code>	style commands 104
..... 69	removed <code>\makefirstuc</code> (now
new 68	dealt with in <code>\glsentryfmt</code>) 104
<code>\gls@writedefhook: new</code> 67	<code>\@Glspl@:</code> add <code>\glsifplural</code> ,
<code>\makeglossaries:</code> Added	<code>\glscapscase</code> , <code>\glscustomtext</code>
providecommand code to aux	and <code>\glsinsert</code> 107
file 154	change to using <code>\glsentryfmt</code>
<code>\new@glossaryentry: new</code> 63	style commands 108
<code>\newglossary: added \@gls@provide@newglossary</code>	removed <code>\makefirstuc</code> (now
..... 54	dealt with in <code>\glsentryfmt</code>) 108
3.11a	<code>\@acrlong:</code> added <code>\glslabel</code> ,
<code>\@ACRlong:</code> added <code>\glslabel</code> ,	<code>\glsifplural</code> , <code>\glscapscase</code> ,
<code>\glsifplural</code> , <code>\glscapscase</code> ,	<code>\glsinsert</code> and <code>\glscustomtext</code>
<code>\glsinsert</code> and <code>\glscustomtext</code> 328
..... 329	<code>\@acrshort:</code> added <code>\glslabel</code> ,
<code>\@ACRshort:</code> added <code>\glslabel</code> ,	<code>\glsifplural</code> , <code>\glscapscase</code> ,
<code>\glsifplural</code> , <code>\glscapscase</code> ,	<code>\glsinsert</code> and <code>\glscustomtext</code>
<code>\glsinsert</code> and <code>\glscustomtext</code> 327
..... 328	<code>\@gls@:</code> add <code>\glslabel</code> ,
<code>\@Acrlong:</code> added <code>\glslabel</code> ,	<code>\glsifplural</code> , <code>\glscapscase</code> ,
<code>\glsifplural</code> , <code>\glscapscase</code> ,	<code>\glscustomtext</code> and
<code>\glsinsert</code> and <code>\glscustomtext</code>	<code>\glsinsert</code> 102
..... 328	change to using <code>\glsentryfmt</code>
<code>\@Acrshort:</code> added <code>\glslabel</code> ,	style commands 102
<code>\glsifplural</code> , <code>\glscapscase</code> ,	<code>\@gls@noexpand@fields:</code> Fixed
<code>\glsinsert</code> and <code>\glscustomtext</code>	bug expand replaced with
..... 327	noexpand 61
<code>\@GLS@:</code> add <code>\glslabel</code> ,	<code>\@glsdisp:</code> add <code>\glslabel</code> ,
<code>\glsifplural</code> , <code>\glscapscase</code> ,	<code>\glsifplural</code> , <code>\glscapscase</code> ,
<code>\glscustomtext</code> and	<code>\glscustomtext</code> and
<code>\glsinsert</code> 105	<code>\glsinsert</code> 110
change to using <code>\glsentryfmt</code>	change to using <code>\glsentryfmt</code>
style commands 105	style commands 110
removed <code>\MakeUppercase</code>	<code>\@glspl@:</code> add <code>\glslabel</code> ,
(now moved to <code>\glsentryfmt</code>)	<code>\glsifplural</code> , <code>\glscapscase</code> ,
..... 105	<code>\glscustomtext</code> and
<code>\@GLSpl:</code> add <code>\glslabel</code> ,	<code>\glsinsert</code> 106
<code>\glsifplural</code> , <code>\glscapscase</code> ,	change to using <code>\glsentryfmt</code>
<code>\glscustomtext</code> and	style commands 106
<code>\glsinsert</code> 109	General: added <code>\glslabel</code> ,
change to using <code>\glsentryfmt</code>	<code>\glsifplural</code> , <code>\glscapscase</code> ,
style commands 109	<code>\glsinsert</code> and <code>\glscustomtext</code>
removed <code>\MakeUppercase</code> 128–137
as now dealt with in	changed to just use <code>\Glsentrydescplural</code>
<code>\glsentryfmt</code> 109 118
<code>\@Gls@:</code> add <code>\glsifplural</code> ,	changed to just use <code>\glsentrydescplural</code>
<code>\glscapscase</code> , <code>\glscustomtext</code> 118, 119
and <code>\glsinsert</code> 103	

changed to just use <code>\Glsentrydesc</code> 117	changed to just use <code>\Glsentryuservi</code> 127
changed to just use <code>\glentrydesc</code> 117, 118	changed to just use <code>\glentryuservi</code> 126, 127
changed to just use <code>\Glsentryfirstplural</code> 115	changed to just use <code>\Glsentryuserv</code> 126
changed to just use <code>\glentryfirstplural</code> 115	changed to just use <code>\glentryuserv</code> 125, 126
changed to just use <code>\Glsentryfirst</code> 113	Now requires textcase 3
changed to just use <code>\glentryfirst</code> 113	acronymlists: replaced <code>\@addtoacronymlists</code> with <code>\DeclareAcronymList</code> 15
changed to just use <code>\Glsentryname</code> 116	<code>\defglssdisplay</code> : obsoleted 88
changed to just use <code>\glentryname</code> 116	<code>\defglssdisplayfirst</code> : obso- leted 88
changed to just use <code>\Glsentryplural</code> 114	<code>\defglssentryfmt</code> : new 53
changed to just use <code>\glentryplural</code> 114	<code>\forglssentries</code> : replaced <code>\ifx</code> with <code>\ifdefempty</code> 47
changed to just use <code>\Glsentrysymbolplural</code> 120	<code>\glss@assign@desc</code> : new 67
changed to just use <code>\glssentrysymbolplural</code> 120, 121	<code>\glss@defglossaryentry</code> : Fixed default counter if none sup- plied 72
changed to just use <code>\Glsentrysymbol</code> 119	<code>\glss@doentryfmt</code> : new 53
changed to just use <code>\glssentrysymbol</code> 119, 120	<code>\glssdisplay</code> : obsoleted 88
Changed to just use <code>\Glsentrytext</code> 112	<code>\glssdisplayfirst</code> : obsoleted .. 87
changed to just use <code>\glssentrytext</code> 111	<code>\glssgenentryfmt</code> : new 82
changed to just use <code>\Glsentryuseriii</code> 124	<code>\glssgetgrouptitle</code> : Added check in case non-Latin alpha- bet in use 189
changed to just use <code>\glssentryuseriii</code> 123, 124	<code>\glssglossarymark</code> : replaced <code>\MakeUppercase</code> with <code>\mfirstucMakeUppercase</code> . 35
changed to just use <code>\Glsentryuserii</code> 123	<code>\glssnavigation</code> : switched to us- ing <code>\@glss@getgrouptitle</code> 248
changed to just use <code>\glssentryuserii</code> 122, 123	<code>\ifglsshasdesc</code> : replaced <code>\ifdefempty</code> with <code>\ifcseempty</code> 50
changed to just use <code>\Glsentryuseriv</code> 125	<code>\ifglsshaslong</code> : new 51
changed to just use <code>\glssentryuseriv</code> 124, 125	<code>\ifglsshasshort</code> : new 51
changed to just use <code>\Glsentryuseri</code> 121	<code>\ifglsshasymbol</code> : replaced <code>\ifdefempty</code> with <code>\ifcseempty</code> 51
changed to just use <code>\glssentryuseri</code> 121, 122	<code>\ifglssused</code> : replaced <code>\ifthenelse</code> with <code>\ifbool</code> 49
	<code>\longnewglossaryentry</code> : new . 67
	<code>\newglossary</code> : replaced <code>\glssdisplay</code> and <code>\glssdisplayfirst</code> with <code>\glssentryfmt</code> 54
	compatible-3.07: cnew 25

\SetCustomDisplayStyle: updated to use \defglentryfmt	229	\glossarysection: changed	
\SetDefaultAcronymDisplayStyle: changed to use \defglentryfmt	214	\glossarymark to \glsglossarymark	35
\SetDescriptionAcronymDisplayStyle: updated to use \defglentryfmt	219	\glossarystyle: fixed bug caused by using \ifdef instead of \ifcsdef	191
\SetDescriptionDUAAcronymDisplayStyle: updated to use \defglentryfmt	218	\gls@assign@desc@field: changed to use \glsetnoexpandfield	17
\SetDescriptionFootnoteAcronymDisplayStyle: updated to use \defglentryfmt	216	\gls@assign@descplural@field: changed to use \glsetnoexpandfield	17
\SetDUADisplayStyle: updated to use \defglentryfmt ..	227	\gls@assign@name@field: changed to use \glsetnoexpandfield	17
\SetFootnoteAcronymDisplayStyle: updated to use \defglentryfmt	222	\gls@assign@type@field: changed to use \glsetexpandfield	17
\SetSmallAcronymDisplayStyle: updated to use \defglentryfmt	224	\gls@checkseeallowed: new ..	58
\setupglossaries: new	27	\glsaddallunused: set default to \@glo@types	146
\showglo long: new	235	\Glsentryfull: changed to use \acrfullformat	143
\showglo short: new	235	\glentryfull: changed to use \acrfullformat	143
numbers: new	25	\Glsentryfullpl: changed to use \acrfullformat	143
symbols: new	25	\glentryfullpl: changed to use \acrfullformat	143
3.12a		\glsglossarymark: renamed \glossarymark to \glsglossarymark to avoid conflict with memoir	35
\gls@defglossaryentry: added \glslabel	68	\glsprestandardsort: new ...	10
\glsaddkey: new	65	\glsetexpandfield: new	17
3.13a		\glsetnoexpandfield: new ..	17
\@gls@assign@symbol@field: changed to use \glsetnoexpandfield	18	altsuper4colheader: switched to \tabularnewline	275
\@gls@assign@symbolplural@field: changed to use \glsetnoexpandfield	18	altsuper4colheaderborder: switched to \tabularnewline	276
\@gls@link: removed \relax ..	91	long: switched to \tabularnewline	254, 255
\@gls@notranslatorhook: new	21	long3col: switched to \tabularnewline	256
\@gls@setupsort@standard: moved \@gls@santizesort to \glsprestandardsort ..	10	long3colheader: switched to \tabularnewline	257
General: added cs@gls@notranslatorhook to else clause	31	long3colheaderborder: switched to \tabularnewline	257
ucmark: added check for memoir .	9		
see: added \gls@checkseeallowed	58		

sm-short-long-desc: new ...	207	\@Acrshort: removed \glslabel (defined in \@gls@link) ...	327
\makeglossaries: made pream- ble only	156	\@GLS@: removed \glslabel (de- fined in \@gls@link)	105
index: new	26	\@GLSpl: removed \glslabel (defined in \@gls@link) ...	109
\newacronymstyle: new	202	\@Gls@: removed \glslabel (de- fined in \@gls@link)	103
long-sc-short: new	204	\@Gls@entry@field: new	138
long-sc-short-desc: new ...	206	\@Glspl@: removed \glslabel (defined in \@gls@link) ...	107
long-short: new	203	\@acrlong: removed \glslabel (defined in \@gls@link) ...	328
long-short-desc: new	205	\@acrshort: removed \glslabel (defined in \@gls@link) ...	327
long-sm-short: new	205	\@gls@: removed \glslabel (de- fined in \@gls@link)	102
long-sm-short-desc: new ...	206	\@gls@access@display: new .	315
footnote: new	209	\@gls@entry@field: new	138
footnote-desc: new	211	\@gls@fetchfield: new	64
footnote-sc: new	211	\@gls@field@link: new	111
footnote-sc-desc: new	211	\@gls@link: added \glsdetoklabel	
footnote-sm: new	211	moved \@gls@link@opts	
footnote-sm-desc: new	212	and \@gls@link@label to	
\setacronymstyle: new	202	\@gls@link	90
\SetDescriptionAcronymDisplayStyle:		\@gls@writedef: added	
Moved check for empty cus- tom text to prevent unwanted		\glsdetoklabel	63
parenthetical material	220	\@glsdisp: removed \glslabel (defined in \@gls@link) ...	110
\SetDescriptionFootnoteAcronymDisplayStyle:		\@Glspl@: removed \glslabel (defined in \@gls@link) ...	106
Moved check for empty cus- tom text to prevent unwanted		\@printglossary: added	
parenthetical material	216	\glsdetoklabel	170
\SetFootnoteAcronymDisplayStyle:		General: changed default to	
Moved check for empty cus- tom text to prevent unwanted		\@empty instead of \relax ..	25
parenthetical material	222	removed \glslabel (defined in	
\SetGenericNewAcronym: new	201	\@gls@link)	128
\SetSmallAcronymDisplayStyle:		sc-short-long-desc: redefined	
Moved check for empty cus- tom text to prevent unwanted		to use accessibility informa- tion	333
parenthetical material	225	\compatibleglossentry: added	
dua: new	207	\glsdetoklabel	310
dua-desc: new	209	\compatiblesubglossentry:	
numberedsection: added		added \glsdetoklabel ...	310
nameref option	6	\Genacrfullformat: redefined	
4.03		to use accessibility informa- tion	326
\@do@wrglossary: added			
\glsdetoklabel	164		
\@ACRlong: removed \glslabel (defined in \@gls@link) ...	329		
\@ACRshort: removed \glslabel (defined in \@gls@link) ...	328		
\@Acrlong: removed \glslabel (defined in \@gls@link) ...	328		

<code>\genacrfullformat:</code> redefined to use accessibility information 326	<code>\glsentrysymbolaccess:</code> switched to using <code>\@gls@entry@field</code> 314
<code>\Genplacrfullformat:</code> redefined to use accessibility information 327	<code>\glsentrysymbolpluralaccess:</code> switched to using <code>\@gls@entry@field</code> 314
<code>\genplacrfullformat:</code> redefined to use accessibility information 327	<code>\glsentrytextaccess:</code> switched to using <code>\@gls@entry@field</code> 314
<code>\glossentryname:</code> added <code>\glsdetoklabel</code> 186	<code>\glsgenacfmt:</code> redefined to use accessibility information ... 324
<code>\gls@defglossaryentry:</code> added <code>\glsdetoklabel</code> 68	<code>\glsgenentryfmt:</code> redefined to use accessibility information 321
replaced <code>#1</code> with <code>\@glo@label</code> 70	<code>\glshyperlink:</code> added <code>\glsdetoklabel</code> 145
replaced <code>\ifthenelse</code> with <code>\ifdefequal</code> 70	<code>\glslocalreset:</code> added <code>\glsdetoklabel</code> 76
<code>\glsadd:</code> added <code>\glsdetoklabel</code> 145	<code>\glslocalunset:</code> added <code>\glsdetoklabel</code> 77
<code>\glsaddkey:</code> switched to using <code>\@gls@field@link</code> 66	<code>\glsmoveentry:</code> added <code>\glsdetoklabel</code> 73
<code>\glsdetoklabel:</code> new 48	replaced <code>\ifthenelse</code> with <code>\ifdefequal</code> 74
<code>\glsdisplaynumberlist:</code> added <code>\glsdetoklabel</code> 144	<code>\glsrefentry:</code> added <code>\glsdetoklabel</code> 184
<code>\glsdoifexistsorwarn:</code> new .. 49	<code>\glsreset:</code> added <code>\glsdetoklabel</code> 76
<code>\glsentryaccess:</code> switched to using <code>\@gls@entry@field</code> . 314	<code>\glsseelist:</code> added <code>\expandafter</code> commands 167
<code>\glsentrydescaccess:</code> switched to using <code>\@gls@entry@field</code> 314	<code>\glsstepentry:</code> added <code>\glsdetoklabel</code> 184
<code>\glsentrydescpluralaccess:</code> switched to using <code>\@gls@entry@field</code> 314	<code>\glsstepsubentry:</code> added <code>\glsdetoklabel</code> 184
<code>\glsentryfirstaccess:</code> switched to using <code>\@gls@entry@field</code> 314	<code>\glsunset:</code> added <code>\glsdetoklabel</code> 76
<code>\glsentryfirstplural:</code> added <code>\glsdetoklabel</code> 140	short-long: commented spuri- ous EOL 204
<code>\glsentrylongaccess:</code> switched to using <code>\@gls@entry@field</code> 315	redefined to use accessibility in- formation 331
<code>\glsentrylongpluralaccess:</code> switched to using <code>\@gls@entry@field</code> 315	short-long-desc: redefined to use accessibility information 333
<code>\glsentrypluralaccess:</code> switched to using <code>\@gls@entry@field</code> 314	<code>\ifglsdescsuppressed:</code> added <code>\glsdetoklabel</code> 50
<code>\glsentryshortaccess:</code> switched to using <code>\@gls@entry@field</code> 315	fixed typo 50
<code>\glsentryshortpluralaccess:</code> switched to using <code>\@gls@entry@field</code> 315	<code>\ifglsentryexists:</code> added <code>\glsdetoklabel</code> 49
	<code>\ifglschaschildren:</code> added <code>\glsdetoklabel</code> 50
	<code>\ifglschasdesc:</code> added <code>\glsdetoklabel</code> 50

\showglotext:added\glsdetoklabel 232	\@glo@sortmacro@nocase: new 176
\showglotextaccess: added \glsdetoklabel 345	\@glo@sortmacro@standard: new 175
\showglotype:added\glsdetoklabel 232	\@glo@sortmacro@use: new .. 177
\showglouserii:added\glsdetoklabel 233	\@glo@sortmacro@word: new . 175
\showglouseriii: added \glsdetoklabel 233	\@gls@getcounterprefix: added warning if no prefix can be formed 165
\showglouseriv: added \glsdetoklabel 233	\@gls@getothergrouptitle: new 190
\showglouservi:added\glsdetoklabel 233	\@gls@noidx@do: new 178
\showglouserii: added \glsdetoklabel 233	\@gls@noref@warn: new 160
\showglouseriii: added \glsdetoklabel 233	\@gls@reference: new 180
\showglouseriv: added \glsdetoklabel 233	\@gls@warnonglossdefined: new 16
\showglouservi: added \glsdetoklabel 234	\@gls@warnontheGLOSSdefined: new 16
dua: fixed bug in \acrfullfmt . 208	\@no@makeglossaries: new .. 160
fixed bug in \Acrfullplfmt . 209	\@print@glossary: new 170
fixed bug in \acrfullplfmt . 209	\@print@noidx@glossary: new 177
redefined to use accessibility in- formation 334	\@print@gloss@setsort: new . 169
dua-desc: commented spurious EOL 209	\@print@glossary: new 169
redefined to use accessibility in- formation 336	General: added sort key to print- gloss group 182
4.04	\compatibleglossentry: changed \newcommand to \def as is may or may not be defined 310
\@gls@noidx@nosanitizesort: new 18	\compatiblesubglossentry: changed \newcommand to \def as is may or may not be defined 310
\@gls@noidx@sanitizesort: new 18	\defglsdisplayfirst: fixed un- wanted space 88
\@gls@nosanitizesort: new . 18	\glo@grabfirst: new 178
\@gls@sanitizesort: new ... 18	\gls@defglossaryentry: re- placed \ifx with \ifdefvoid 73
\@glo@addchildren: new 172	\glsnoidxdisplayloc: new .. 180
\@glo@do@sortentries: new . 173	\glsnoidxdisplayloclisthandler: new 180
\@glo@grabfirst: new 178	\glsnoidxloclist: new 179
\@glo@sortedinsert: new ... 173	\glsnoidxloclisthandler: new 180
\@glo@sortentries: new 172	\glsnoidxstripaccents: new . 19
\@glo@sorthandler@case: new 174	alttree: moved hangindent and parindent assignments out- side level test 287
\@glo@sorthandler@letter: new 174	\makeglossaries: Moved def- inition of \glswrite to
\@glo@sorthandler@nocase: new 174	
\@glo@sorthandler@word: new 174	
\@glo@sortmacro@case: new . 175	
\@glo@sortmacro@def: new .. 176	
\@glo@sortmacro@def@do: new 176	
\@glo@sortmacro@letter: new 175	

\makeglossaries	154	\@Glspl@: added \glsifhyper	107
\makenoidxglossaries: new .	156	\@gls@: added \glsifhyper ..	102
\printglossary: changed to use		\@gls@numbersdef: added hook	
new \@printglossary ...	168	to set toc title	26
\printnoidxglossaries: new	169	\@gls@symbolsdef: added hook	
\printnoidxglossary: new ..	168	to set toc title	25
\showgloclist: new	235	\@glsdisp: added \glsifhyper	110
\warn@noprintglossary: Acti-		\@Glspl@: added \glsifhyper	106
vate warning in \makeglossaries		General: added \glsifhyper ..	
.....	168	128–138
\writeist: checked for definition		acronym: added hook to set toc ti-	
of \glswrite	148, 152	tle	13
4.06		acronyms: added hook to set toc	
\@GLS@: added \glsifhyper ..	105	title	14
\@GLSpl: added \glsifhyper .	109	\glsdefmain: added hook to set	
\@Gls@: added \glsifhyper ..	103	toc title	12

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	164	\@glo@default@sorttype	9
\@do@wrglossary	163	\@glo@do@sortentries	173
\@glo@assign@sortkey	182	\@glo@grabfirst	178
\@glossarysec	5	\@glo@no@assign@sortkey	182
\@glossaryseclabel	6	\@glo@seeautonumberlist	7
\@glossarysecstar	6	\@glo@sortedinsert	173
\@gls@default@entryfmt	317	\@glo@sortentries	172
\@gls@expand@field	61	\@glo@sorthandler@case	174
\@gls@fixbraces	166	\@glo@sorthandler@letter	174
\@gls@noidx@nosanitizesort	18	\@glo@sorthandler@nocase	174
\@gls@noidx@sanitizesort	18	\@glo@sorthandler@word	174
\@gls@nosanitizesort	18	\@glo@sortmacro@case	175
\@gls@sanitizesort	18	\@glo@sortmacro@def	176
\@ACRlong	329	\@glo@sortmacro@def@do	176
\@ACRshort	328	\@glo@sortmacro@letter	175
\@Acrlong	328	\@glo@sortmacro@nocase	176
\@Acrshort	327	\@glo@sortmacro@standard	175
\@GLS@	105	\@glo@sortmacro@use	177
\@GLSpl	109	\@glo@sortmacro@word	175
\@Gls@	103	\@glo@storeentry	74
\@Gls@entry@field	138	\@glo@types	53
\@Glspl@	107	\@glossary	161
\@PGLS	243	\@glossary@default@style	6
\@PGLS@	243	\@glossaryentryfield	74
\@PGLSpl	243	\@glossarysection	36
\@PGLSpl@	243	\@glossarysubentryfield	74
\@Pgls	241	\@gls	102
\@Pgls@	241	\@gls@	102
\@Pglspl	242	\@gls@@link	90
\@Pglspl@	242	\@gls@access@display	315
\@acrfull	197	\@gls@addpredefinedattributes	41
\@acrlong	328		
\@acrshort	327	\@gls@assign@symbol@field	18
\@addtoacronymlists	14	\@gls@assign@symbolplural@field	18
\@delimN	193		
\@delimR	193	\@gls@checkactual	98
\@disable@onlypremakeg	28	\@gls@checkbar	97
\@disable@premakecs	28	\@gls@checkescactual	96
\@disabled@glsaddxdycounters	40	\@gls@checkescbar	96
\@do@seeglossary	165	\@gls@checkesclevel	97
\@do@wrglossary	162, 291	\@gls@checkescquote	95
\@glo@addchildren	172	\@gls@checklevel	98
		\@gls@checkmkidxchars	94

<code>\@gls@checkquote</code>	95	<code>\@gls@setupsort@use</code>	11
<code>\@gls@codepage</code>	46	<code>\@gls@startswithexpandonce</code>	62
<code>\@gls@counterwithin</code>	9	<code>\@gls@symbolsdef</code>	25
<code>\@gls@declareoption</code>	7	<code>\@gls@tmpb</code>	95
<code>\@gls@default@value</code>	57	<code>\@gls@toc</code>	38
<code>\@gls@do@acronymsdef</code>	13	<code>\@gls@updatechecked</code>	95
<code>\@gls@entry@field</code>	138	<code>\@gls@warnonglossdefined</code>	16
<code>\@gls@escbsdq</code>	93	<code>\@gls@warnontheGLOSSdefined</code>	16
<code>\@gls@expand@fields</code>	61	<code>\@gls@writedef</code>	63
<code>\@gls@fetchfield</code>	64	<code>\@gls@xdy@Lclass@Alpha-page-numbers</code>	
<code>\@gls@field@link</code>	111	43
<code>\@gls@fixbraces</code>	166	<code>\@gls@xdy@Lclass@Appendix-page-numbers</code>	
<code>\@gls@getcounter</code>	55	43
<code>\@gls@getcounterprefix</code>	165	<code>\@gls@xdy@Lclass@Roman-page-numbers</code>	
<code>\@gls@getgrouptitle</code>	189	42
<code>\@gls@getothergrouptitle</code>	190	<code>\@gls@xdy@Lclass@alpha-page-numbers</code>	
<code>\@gls@hypergroup</code>	247	43
<code>\@gls@ifinlist</code>	39	<code>\@gls@xdy@Lclass@arabic-page-numbers</code>	
<code>\@gls@indexdef</code>	26	43
<code>\@gls@keymap</code>	64, 238	<code>\@gls@xdy@Lclass@arabic-section-numbers</code>	
<code>\@gls@link</code>	90	43
<code>\@gls@loadlist</code>	8	<code>\@gls@xdy@Lclass@roman-page-numbers</code>	
<code>\@gls@loadlong</code>	8	42
<code>\@gls@loadsuper</code>	8	<code>\@gls@xdy@locationlist</code>	42
<code>\@gls@loadtree</code>	8	<code>\@gls@xdy@checkbackslash</code>	99
<code>\@gls@makefirsttuc</code>	245	<code>\@gls@xdy@checkquote</code>	99
<code>\@gls@missingnumberlist</code>	60	<code>\@gls@Alphacompositor</code>	33, 43
<code>\@gls@noaccess</code>	312	<code>\@gls@acronymlists</code>	14
<code>\@gls@noexpand@field</code>	60	<code>\@gls@defaultplural</code>	60
<code>\@gls@noexpand@fields</code>	60	<code>\@gls@defaultsort</code>	60
<code>\@gls@nohyperlist</code>	16	<code>\@gls@disp</code>	110
<code>\@gls@noidx@do</code>	178	<code>\@gls@firstletter</code>	147
<code>\@gls@noidx@setsanitizesort</code>	20	<code>\@gls@hypernumber</code>	193
<code>\@gls@noref@warn</code>	160	<code>\@gls@link</code>	100
<code>\@gls@notranslatorhook</code>	21	<code>\@gls@minrange</code>	147
<code>\@gls@numbersdef</code>	26	<code>\@gls@nextpages</code>	183
<code>\@gls@onlypremakeg</code>	28	<code>\@gls@nodesc</code>	60
<code>\@gls@provide@newglossary</code>	53	<code>\@gls@noname</code>	59
<code>\@gls@reference</code>	180	<code>\@gls@nonextpages</code>	182
<code>\@gls@renewglossary</code>	161	<code>\@gls@openfile</code>	154
<code>\@gls@sanitizedesc</code>	17	<code>\@gls@order</code>	23
<code>\@gls@sanitizename</code>	17	<code>\@gls@spl@</code>	106
<code>\@gls@sanitizesort</code>	18	<code>\@gls@target</code>	100
<code>\@gls@sanitizesymbol</code>	18	<code>\@gls@widestname</code>	287
<code>\@gls@saveentrycounter</code>	91	<code>\@gls@writefiles</code>	160
<code>\@gls@setacrstyle</code>	22	<code>\@istfilename</code>	32
<code>\@gls@setcounter</code>	55	<code>\@makeglossary</code>	153
<code>\@gls@setupsort@def</code>	11	<code>\@newglossary</code>	55
<code>\@gls@setupsort@standard</code>	10	<code>\@newglossaryentryposthook</code>	73

<code>\@newglossaryentryprehook</code>	73
<code>\@no@makeglossaries</code>	160
<code>\@no@post@desc</code>	32
<code>\@nopostdesc</code>	32
<code>\@onlypremakeg</code>	28
<code>\@p@glossarysection</code>	37
<code>\@pgls</code>	240
<code>\@pgls@</code>	240
<code>\@pglsp1</code>	240
<code>\@pglsp1@</code>	240
<code>\@print@glossary</code>	170
<code>\@print@noidx@glossary</code>	177
<code>\@printgloss@setsort</code>	169
<code>\@printglossary</code>	169
<code>\@sPGLS</code>	242
<code>\@sPGLSp1</code>	243
<code>\@sPgls</code>	241
<code>\@sPglspl</code>	242
<code>\@set@glo@numformat</code>	92, 292
<code>\@sgls</code>	102, 110
<code>\@sgls@link</code>	90
<code>\@spgls</code>	240
<code>\@spglsp1</code>	240
<code>\@wrglossary</code>	162
<code>\@xdy@main@language</code>	24
<code>\@xdy@attributelist</code>	39
<code>\@xdy@attributes</code>	38
<code>\@xdy@language</code>	46
<code>\@xdylettergroups</code>	47
<code>\@xdylocationclassorder</code>	44
<code>\@xdylocref</code>	39
<code>\@xdyrequiredstyles</code>	45
<code>\@xdysortrules</code>	45
<code>\@xdyuseralphabets</code>	41
<code>\@xdyuserlocationdefs</code>	43
<code>\@xdyuserlocationnames</code>	43

A

<code>\Ac</code>	213
<code>\ac</code>	213
<code>access (key)</code>	310
<code>accsupp package</code>	309
<code>\accsuppglossaryentryfield</code>	330
<code>\accsuppglossarysubentryfield</code>	330
<code>\Acf</code>	213
<code>\acf</code>	213
<code>\Acfp</code>	213
<code>\acfp</code>	213

<code>\Acl</code>	213
<code>\acl</code>	212
<code>\Aclp</code>	213
<code>\aclp</code>	213
<code>\Acp</code>	213
<code>\acp</code>	213
<code>\acrfootnote</code>	215
<code>\ACRfull</code>	198
<code>\Acrfull</code>	197
<code>\acrfull</code>	196, 197, 201, 210, 337
<code>\ACRfullfmt</code>	198
<code>\Acrfullfmt</code>	198
<code>\acrfullfmt</code>	197
<code>\acrfullformat</code>	85, 197
<code>\ACRfullpl</code>	199
<code>\Acrfullpl</code>	199
<code>\acrfullpl</code>	198
<code>\ACRfullplfmt</code>	200
<code>\Acrfullplfmt</code>	199
<code>\acrfullplfmt</code>	199
<code>\acrlinkfootnote</code>	215
<code>\acrlinkfullformat</code>	197
<code>\ACRlong</code>	134
<code>\Acrlong</code>	133
<code>\acrlong</code>	132
<code>\ACRlongpl</code>	137
<code>\Acrlongpl</code>	136
<code>\acrlongpl</code>	135
<code>\acrnameformat</code>	200, 221
<code>\acrno linkfootnote</code>	215
<code>acronym (option)</code>	13

acronym styles:

<code>dua</code>	207, 334
<code>dua-desc</code>	209, 336
<code>footnote</code>	209, 336
<code>footnote-desc</code>	211, 339
<code>footnote-sc</code>	211, 338
<code>footnote-sc-desc</code>	211, 339
<code>footnote-sm</code>	211, 338
<code>footnote-sm-desc</code>	212, 339
<code>long-sc-short</code>	204
<code>long-sc-short-desc</code>	206, 332
<code>long-short</code>	203, 330
<code>long-short-desc</code>	205, 332
<code>long-sm-short</code>	205
<code>long-sm-short-desc</code>	206, 333
<code>sc-short-long</code>	205
<code>sc-short-long-desc</code>	207, 333
<code>short-long</code>	204, 331

short-long-desc 206, 333
 sm-short-long 205
 sm-short-long-desc .. 207, 334
 \acronymentry 202
 \acronymfont
 85, 200, 217, 221, 224, 227
 acronymlists (option) 15
 \acronymname 29
 acronyms (option) 14
 \acronymsort 202
 \acronymstype 13, 195
 \acrpluralsuffix 196
 \ACRshort 129
 \Acrshort 128
 \acrshort 127
 \ACRshortpl 132
 \Acrshortpl 131
 \acrshortpl 130
 \Acs 212
 \acs 212
 \Acsp 212
 \acsp 212
 \addglossarytocaptions 30
 \addto 30
 align (environment) 76, 91
 altlist (style) 252
 altlistgroup (style) 252
 altlisthypergroup (style) 253
 altlong4col (style) 259
 altlong4colborder (style) 259
 altlong4colheader (style) 259
 altlong4colheaderborder (style) 259
 altlongragged4col (style) 263
 altlongragged4colborder (style) 264
 altlongragged4colheader (style) 264
 altlongragged4colheaderborder
 (style) 265
 \altnewglossary 54
 altsuper4col (style) 275
 altsuper4colborder (style) 275
 altsuper4colheader (style) 275
 altsuper4colheaderborder
 (style) 275
 altsuperragged4col (style) 280
 altsuperragged4colborder
 (style) 281
 altsuperragged4colheader
 (style) 281

altsuperragged4colheaderborder
 (style) 281
 alttree (style) 287
 alttreegroup (style) 289
 alttreehypergroup (style) 289
 amsgen package 4, 89
 amsmath package 76
 \andname 30
 array package 260, 276
 article class 165

B

babel package 27, 29, 30, 46, 346

C

\capitalisewords 245
 \changes 5
 \compatglossarystyle 296
 compatible-2.07 (option) 25
 compatible-3.07 (option) 25
 \compatibleglossentry .. 186, 310
 \compatiblesubglossentry 187, 310
 counter (key) 57
 counter (option) 15
 \CustomAcronymFields 230
 \CustomNewAcronymDef 230

D

datatool package 174
 \DeclareAcronymList 14
 \DefaultNewAcronymDef .. 214, 340
 \defentryfmt 89
 \defglstdisplay 88
 \defglstdisplayfirst 88
 \defglssentry 54
 \defglssentryfmt 53, 56, 57, 79
 \DefineAcronymSynonyms 212
 \delimN 34, 192
 \delimR 34, 192
 description (environment) 251
 description (key) 56
 description (option) 22
 descriptionaccess (key) 311
 \DescriptionDUANewAcronymDef 218
 \DescriptionFootnoteNewAcronymDef
 216, 341
 \descriptionname 29
 \DescriptionNewAcronymDef ..
 220, 342
 descriptionplural (key) 56

descriptionpluralaccess (key) 311
\detokenize 48
doc package 4, 5, 12
dua (acrstyle) 207, 334
dua (option) 23
dua-desc (acrstyle) 209, 336
\DUANewAcronymDef 227

E

entrycounter (option) 9
entrycounterwithin (option) 9
\entryname 29
environments:
 align 76, 91
 description 251
 longtable 8, 231, 254–265
 multicols 266
 supertabular ... 8, 231, 269–282
 theglossary ... 5, 16, 34, 35,
 185, 186, 191, 267, 268, 284–287
 theindex 282
equation (counter) 91, 92
etoolbox package 4, 244

F

file types
 .aux 171
 .glo 74
 .ist 146, 147, 153
 .toc 38
 .xdy 32
 glo 237
\findrootlanguage 45
first (key) 57
firstaccess (key) 311
\firstacronymfont 85, 200
firstplural (key) 57
firstpluralaccess (key) 311
footnote (acrstyle) 209, 336
footnote (option) 22
footnote-desc (acrstyle) .. 211, 339
footnote-sc (acrstyle) 211, 338
footnote-sc-desc (acrstyle) 211, 339
footnote-sm (acrstyle) 211, 338
footnote-sm-desc (acrstyle) 212, 339
\FootnoteNewAcronymDef . 222, 343
\forallglossaries 47
\forallglsentries 48
\forallglsentries 47

G

garamondx package 196
\Genacrfullformat 87, 326
\genacrfullformat 87, 326
\GenericAcronymFields 202
\Genplacrfullformat 87, 327
\genplacrfullformat 87, 327
\glo@grabfirst 178
\glolinkprefix 91
glossareentry (counter) 184
glossaries package 26,
 46, 147, 231, 237, 251, 289, 309
glossaries-accsupp package ... 74, 309
\GlossariesWarning 16
\GlossariesWarningNoLine 16
\glossary 53, 153, 161, 190
glossary counters:
 glossaryentry 183
 glossarysubentry 183
glossary keys:
 access 310
 counter 57
 description 56
 descriptionaccess 311
 descriptionplural 56
 descriptionpluralaccess . 311
 first 57
 firstaccess 311
 firstplural 57
 firstpluralaccess 311
 long 59
 longaccess 312
 longplural 59
 longpluralaccess 312
 name 56
 nonumberlist 58
 parent 58
 plural 56
 pluralaccess 311
 see 58
 short 59
 shortaccess 311
 shortplural 59
 shortpluralaccess 311
 sort 56
 symbol 57
 symbolaccess 311
 symbolplural 57
 symbolpluralaccess 311

text	56	altsuperragged4colborder	
textaccess	311	281, 307
type	57	altsuperragged4colborder	281
user1	58	altsuperragged4colheader	
user2	58	281, 307
user3	59	altsuperragged4colheader	281
user4	59	altsuperragged4colheaderborder	
user5	59	281, 307
user6	59	altsuperragged4colheaderborder	
glossary package	1, 195	281
glossary styles:		alttree ...	268, 286, 287, 289, 304
altlist	252, 253, 297	alttree	287
altlist	252	alttreegroup	289, 305
altlistgroup	252, 253, 298	alttreegroup	289
altlistgroup	252	alttreehypergroup ...	289, 305
altlisthypergroup ...	253, 298	alttreehypergroup	289
altlisthypergroup	253	index	266, 282–284, 301
altlong4col	259, 263, 300	index	282
altlong4col	259	indexgroup	283, 302
altlong4colborder ...	259, 300	indexgroup	283
altlong4colborder	259	indexhypergroup	283, 302
altlong4colheader ...	259, 300	indexhypergroup	283
altlong4colheader	259	inline	296
altlong4colheaderborder .		inline	248
.....	259, 300	list	6, 251–253, 297
altlong4colheaderborder .	259	list	251
altlongragged4col	263, 264, 301	listdotted	253, 298
altlongragged4col	263	listdotted	253
altlongragged4colborder .		listgroup	251, 252, 297
.....	264, 301	listgroup	251
altlongragged4colborder .	264	listhypergroup	252, 297
altlongragged4colheader .		listhypergroup	252
.....	264, 301	long	254–256, 260, 298, 300
altlongragged4colheader .	264	long	254
altlongragged4colheaderborder		long3col	256, 299
.....	265, 301	long3col	256
altlongragged4colheaderborder		long3colborder	256, 299
.....	265	long3colborder	256
altsuper4col	275, 280, 309	long3colheader	256, 299
altsuper4col	275	long3colheader	256
altsuper4colborder ..	275, 309	long3colheaderborder	257, 299
altsuper4colborder	275	long3colheaderborder ...	257
altsuper4colheader ..	275, 309	long4col	257–259, 299
altsuper4colheader	275	long4col	257
altsuper4colheaderborder		long4colborder	258, 299
.....	275, 309	long4colborder	258
altsuper4colheaderborder	275	long4colheader	258, 299
altsuperragged4col	280, 281, 307	long4colheader	258
altsuperragged4col	280	long4colheaderborder	258, 300

long4colheaderborder	258
longborder	255, 298
longborder	255
longheader	255, 298
longheader	255
longheaderborder	255, 299
longheaderborder	255
longragged	260–262
longragged	260
longragged3col	...	262, 263, 301
longragged3col	262
longragged3colborder		262, 301
longragged3colborder	262
longragged3colheader		263, 301
longragged3colheader	263
longragged3colheaderborder		
	263, 301
longragged3colheaderborder		
	263
longraggedborder	261, 300
longraggedborder	261
longraggedheader	261, 300
longraggedheader	261
longraggedheaderborder		261, 300
longraggedheaderborder	..	261
mcolalttree	268, 306
mcolalttree	268
mcolalttreegroup	269, 306
mcolalttreegroup	268
mcolalttreehypergroup		269, 306
mcolalttreehypergroup	...	269
mcolindex	266, 305
mcolindex	266
mcolindexgroup	266, 305
mcolindexgroup	266
mcolindexhypergroup	.	266, 305
mcolindexhypergroup	266
mcoltree	267, 305
mcoltree	266
mcoltreegroup	305
mcoltreegroup	267
mcoltreehypergroup	..	267, 305
mcoltreehypergroup	267
mcoltreenoname	267, 305
mcoltreenoname	267
mcoltreenonamegroup	.	268, 305
mcoltreenonamegroup	267
mcoltreenonamehypergroup		
	268, 306
mcoltreenonamehypergroup		
	268
sublistdotted	298
sublistdotted	253
super	269–271, 277, 307
super	269
super3col	271, 272, 308
super3col	271
super3colborder	272, 308
super3colborder	272
super3colheader	272, 308
super3colheader	272
super3colheaderborder		272, 308
super3colheaderborder	...	272
super4col	273–275, 308
super4col	273
super4colborder	274, 309
super4colborder	274
super4colheader	274, 309
super4colheader	274
super4colheaderborder		274, 309
super4colheaderborder	...	274
superborder	270, 308
superborder	270
superheader	270, 308
superheader	270
superheaderborder	...	271, 308
superheaderborder	271
superragged	276, 278, 306
superragged	276
superragged3col	..	278–280, 306
superragged3col	278
superragged3colborder		279, 307
superragged3colborder	...	279
superragged3colheader		279, 307
superragged3colheader	...	279
superragged3colheaderborder		
	280, 307
superraggedborder	...	277, 306
superraggedborder	277
superraggedheader	...	277, 306
superraggedheader	277
superraggedheaderborder	.	
	278, 306
superraggedheaderborder	.	278
superraggedright3colheaderborder		
	280
tree	266, 284, 285, 287, 302
tree	284
treegroup	267, 285, 303

treegroup	284	\gls@assign@desc@field	17
treehypergroup	285, 303	\gls@assign@descplural@field	17
treehypergroup	285	\gls@assign@field	62
treenoname ...	267, 285, 286, 303	\gls@assign@name@field	17
treenoname	285	\gls@assign@type@field	17
treenonamegroup	286, 303	\gls@checkisacronymlist	15
treenonamegroup	286	\gls@checkseeallowed	58
treenonamehypergroup	286, 303	\gls@codepage	24
treenonamehypergroup	286	\gls@defglossaryentry	68
glossary-hypernav package	147	\gls@disablepagerefexpansion	162
glossary-list package	6, 8, 251	\gls@docclearpage	37
glossary-long package ..	8, 254, 263, 269	\gls@doentryfmt	53
glossary-longragged package	260	\gls@hypergroup prerun	247
glossary-mcols package	265	\gls@ifnotmeasuring	76
glossary-super package		\gls@level	60
.....	8, 254, 269, 276, 280	\gls@noidxglossary	160
glossary-superragged package	276	\gls@numberpage	163
glossary-tree package	8, 282	\gls@protected@pagefmts	162
glossaryentry (counter) ..	9, 184, 185	\gls@Romanpage	163
glossaryentry (counter)	183	\gls@romanpage	163
\glossaryentryfield ..	188, 191, 192	\gls@save@numberlist	167
\glossaryentrynumber ...	182, 183	\gls@suffixF	33
\glossaryentrynumbers		\gls@suffixFF	33
.....	7, 34, 169, 170	\glsaccessdisplay	317
\glossaryheader	186, 191	\glsaccsupp	315
\glossarymark	36	\glsadd	78, 145, 190
\glossaryname	29, 30	\glsadd options	
\glossarypostamble	35, 191	counter	145
\glossarypreamble	34, 191	format	145, 192
\glossarysection	6, 35, 53	\glsaddall	48, 78, 146
\glossarystyle	191, 231	\glsaddall options	
glossarysubentry (counter) ...		types	145, 146
.....	9, 184, 185	\glsaddallunused	146
glossarysubentry (counter) ...	183	\glsaddkey	65
\glossarysubentryfield	188	\GlsAddLetterGroup	47
\glossentry	57, 186	\GlsAddSortRule	45
\Glossentrydesc	187	\GlsAddXdyAlphabet	41
\glossentrydesc	187, 329	\GlsAddXdyAttribute	40, 290
\Glossentryname	186	\GlsAddXdyCounters	39, 290
\glossentryname	186, 329	\GlsAddXdyLocation	43, 291
\Glossentrysymbol	187	\GlsAddXdyStyle	45
\glossentrysymbol	187, 329	\glsautoprefix	6
\GLS	104	\glsclearpage	38
\Gls	103, 107, 244	\glsclsebrace	147
\gls	4, 57, 78, 89, 101, 104,	\glscompositor	33, 43
106, 111–115, 117–126, 184, 239		\glscounter	15, 54
\gls@Alphpage	163	\GlsDeclareNoHyperList	16
\gls@alphpage	163	\glsdefaulttype	13
\gls@assign@desc	67	\glsdefmain	12

<code>\GLSdesc</code>	117	<code>\glstentryname</code>	139, 167
<code>\Glsdesc</code>	117	<code>\glstentrynumberlist</code>	144, 158
<code>\glsdesc</code>	117	<code>\Glstentryplural</code>	140
<code>\GLSdescplural</code>	118	<code>\glstentryplural</code>	139
<code>\Glsdescplural</code>	118	<code>\glstentrypluralaccess</code>	314
<code>\glsdescplural</code>	118	<code>\Glstentryprefix</code>	239
<code>\glsdescriptionaccessdisplay</code>	316	<code>\glstentryprefix</code>	238
<code>\glsdescriptionpluralaccessdisplay</code>	316	<code>\Glstentryprefixfirst</code>	238
<code>\glsdescwidth</code>	254, 260, 269, 276	<code>\glstentryprefixfirst</code>	238
<code>\glsdetoklabel</code>	48	<code>\Glstentryprefixfirstplural</code>	239
<code>\glsdisablehyper</code>	101	<code>\glstentryprefixfirstplural</code>	238
<code>\glsdisp</code>	110	<code>\Glstentryprefixplural</code>	239
<code>\glsdisplay</code>	78, 88, 101	<code>\glstentryprefixplural</code>	238
<code>\glsdisplayfirst</code>	78, 87, 101	<code>\Glstentryshort</code>	142
<code>\glsdisplaynumberlist</code>	144, 158, 180	<code>\glstentryshort</code>	142
<code>\glsdoifexists</code>	49	<code>\glstentryshortaccess</code>	315
<code>\glsdoifexistsorwarn</code>	49	<code>\Glstentryshortpl</code>	143
<code>\glsdoifnoexists</code>	49	<code>\glstentryshortpl</code>	143
<code>\glsdoparenifnotempty</code>	224	<code>\glstentryshortpluralaccess</code>	315
<code>\glsenablehyper</code>	101	<code>\glstentrysort</code>	141
<code>\glstentryaccess</code>	314	<code>\Glstentrysymbol</code>	140
<code>\glstentrycounter</code>	91	<code>\glstentrysymbol</code>	140
<code>\glstentrycounterlabel</code>	185	<code>\glstentrysymbolaccess</code>	314
<code>\Glstentrydesc</code>	139	<code>\Glstentrysymbolplural</code>	140
<code>\glstentrydesc</code>	139	<code>\glstentrysymbolplural</code>	140
<code>\glstentrydescaccess</code>	314	<code>\glstentrysymbolpluralaccess</code>	314
<code>\Glstentrydescplural</code>	139	<code>\Glstentrytext</code>	139
<code>\glstentrydescplural</code>	139	<code>\glstentrytext</code>	48, 139, 167
<code>\glstentrydescpluralaccess</code>	314	<code>\glstentrytextaccess</code>	314
<code>\Glstentryfirst</code>	140	<code>\glstentrytype</code>	141
<code>\glstentryfirst</code>	140	<code>\Glstentryuseri</code>	141
<code>\glstentryfirstaccess</code>	314	<code>\glstentryuseri</code>	141
<code>\Glstentryfirstplural</code>	141	<code>\Glstentryuserii</code>	141
<code>\glstentryfirstplural</code>	140	<code>\glstentryuserii</code>	141
<code>\glstentryfirstpluralaccess</code>	314	<code>\Glstentryuseriii</code>	142
<code>\glstentryfmt</code>	56, 57, 79	<code>\glstentryuseriii</code>	141
<code>\Glstentryfull</code>	143	<code>\Glstentryuseriv</code>	142
<code>\glstentryfull</code>	143, 201, 210, 338	<code>\glstentryuseriv</code>	142
<code>\Glstentryfullpl</code>	143	<code>\Glstentryuserv</code>	142
<code>\glstentryfullpl</code>	143	<code>\glstentryuserv</code>	142
<code>\glstentryitem</code>	185	<code>\Glstentryuservi</code>	142
<code>\Glstentrylong</code>	143	<code>\glstentryuservi</code>	142
<code>\glstentrylong</code>	143	<code>\glsexpandfields</code>	62
<code>\glstentrylongaccess</code>	315	<code>\GLSfirst</code>	113
<code>\Glstentrylongpl</code>	143	<code>\Glsfirst</code>	113
<code>\glstentrylongpl</code>	143	<code>\glsfirst</code>	112, 113
<code>\glstentrylongplpluralaccess</code>	315	<code>\glsfirstaccessdisplay</code>	316
<code>\Glstentryname</code>	139	<code>\GLSfirstplural</code>	115
		<code>\Glsfirstplural</code>	115

<code>\glsfirstplural</code>	114, 115	<code>\glsname</code>	116
<code>\glsfirstpluralaccessdisplay</code>	316	<code>\glsnameaccessdisplay</code>	316
<code>\glsgenacfmt</code>	85, 324	<code>\glsnamefont</code>	192
<code>\glsgenentryfmt</code>	82, 321	<code>\glsnavhyperlink</code>	246
<code>\glsgetgrouplabel</code>	190	<code>\glsnavhypertarget</code>	247
<code>\glsgetgrouptitle</code>	147, 189	<code>\glsnavigation</code>	248
<code>\glsglossarymark</code>	9, 35	<code>\glsnextpages</code>	183
<code>\glsgroupheading</code>	189, 191	<code>\glsnoexpandfields</code>	62
<code>\glsgroupskip</code>	189, 191, 251	<code>\glsnoidxdisplayloc</code>	180
<code>\glshyperlink</code>	145	<code>\glsnoidxdisplayloclisthandler</code>	180
<code>\glshypernavsep</code>	248	<code>\glsnoidxloclist</code>	179
<code>\glshypernumber</code>	34, 192	<code>\glsnoidxloclisthandler</code>	180
<code>\glsIfListOfAcronyms</code>	14	<code>\glsnoidxnumberlistloophandler</code>	159
<code>\glsinlinedescformat</code>	250	<code>\glsnoidxstripaccents</code>	19
<code>\glsinlinedopostchild</code> ..	249, 250	<code>\glsnonextpages</code>	183
<code>\glsinlineemptydescformat</code> ..	250	<code>\glsnoxindywarning</code>	38
<code>\glsinlineformat</code>	250	<code>\glsnumberformat</code>	34
<code>\glsinlineparentchildseparator</code>	250	<code>\glsnumberlistloop</code>	159
<code>\glsinlinepostchild</code>	250	<code>\glsnumbersgroupname</code> .	29, 147, 189
<code>\glsinlineseparator</code>	250	<code>\glsnumlistlastsep</code>	145
<code>\glsinlinesubdescformat</code>	250	<code>\glsnumlistsep</code>	145
<code>\glsinlinesubnameformat</code>	250	<code>\glsopenbrace</code>	147
<code>\glsinlinesubseparator</code>	250	<code>\glsorder</code>	23
<code>\glskeylisttok</code>	200	<code>\glsorg@endtheglossary</code>	5
<code>\glslabeltok</code>	200	<code>\glsorg@glossary</code>	4
<code>\glslink</code> ...	78, 90, 101, 145, 190, 192	<code>\glsorg@theglossary</code>	5
<code>\glslink options</code>		<code>\glsorg@wrglossary</code>	4
counter	89, 101, 237	<code>\glspagelistwidth</code>	254, 260, 269, 276
format	89, 101, 192	<code>\glspar</code>	32
hyper	89, 101	<code>\GLSpl</code>	108
local	89	<code>\Glspl</code>	107, 244
<code>\glslistdottedwidth</code>	253	<code>\glspl</code>	78, 106–108
<code>\glslocalreset</code>	76	<code>\GLSplural</code>	114
<code>\glslocalresetall</code>	77	<code>\Glsplural</code>	114
<code>\glslocalunset</code>	77	<code>\glsplural</code>	113, 114
<code>\glslocalunsetall</code>	78	<code>\glspluralaccessdisplay</code>	316
<code>\glslongaccessdisplay</code>	317	<code>\glspluralsuffix</code>	30, 56, 57
<code>\glslongaccesskey</code>	345	<code>\glspostdescription</code>	8
<code>\glslongkey</code>	196	<code>\glspostinline</code>	250
<code>\glslongpluralaccessdisplay</code>	317	<code>\glsprestandardsort</code>	10
<code>\glslongpluralaccesskey</code>	345	<code>\glsquote</code>	147
<code>\glslongpluralkey</code>	196	<code>\glsrefentry</code>	184
<code>\glslongtok</code>	200	<code>\glsreset</code>	76
<code>\glsmakefirsttuc</code>	245	<code>\glsresetall</code>	77
<code>\glsmcols</code>	265	<code>\glsresetentrylist</code>	183
<code>\glsmoveentry</code>	73	<code>\glsresetsubentrycounter</code> ...	184
<code>\GLSname</code>	116	<code>\glssee</code>	166
<code>\Glsname</code>	116		

\glsseeformat	149, 166
\glsseeitem	167
\glsseeitemformat	167
\glsseelastsep	167
\glsseelist	166
\glsseesep	167
\glsSetAlphaCompositor	33
\glsSetCompositor	32, 33
\glssetexpandfield	17
\glssetnoexpandfield	17
\glsSetSuffixF	33
\glsSetSuffixFF	34
\glssettoctitle	29
\glssetwidest	286
\GlsSetXdyCodePage	46
\GlsSetXdyFirstLetterAfterDigits	147
\GlsSetXdyLanguage	46
\GlsSetXdyLocationClassOrder	44
\GlsSetXdyMinRangeLength	147
\GlsSetXdyStyles	45
\glsshortaccessdisplay	317
\glsshortaccesskey	345
\glsshortkey	196
\glsshortpluralaccessdisplay	317
\glsshortpluralaccesskey	345
\glsshortpluralkey	196
\glsshorttok	200
\glssortnumberfmt	10
\glsstepentry	184
\glsstepsubentry	184
\glssubentrycounterlabel	185
\glssubentryitem	185
\GLSsymbol	119
\Glsymbol	119
\glssymbol	119
\glssymbolaccessdisplay	316
\glssymbolnav	248
\GLSsymbolplural	121
\Glsymbolplural	120
\glssymbolplural	120
\glssymbolpluralaccessdisplay	316
\glssymbolsgroupname	29, 147, 189
\glstarget	186
\GLStext	112
\Glstext	112
\glstext	111
\glstextaccessdisplay	316

`\ifglstryexists` 48
`\ifglshaschildren` 50
`\ifglshasdesc` 50
`\ifglshasfield` 51
`\ifglshaslong` 51
`\ifglshasparent` 50
`\ifglshasprefix` 239
`\ifglshasprefixfirst` 239
`\ifglshasprefixfirstplural` . 239
`\ifglshasprefixplural` 239
`\ifglshasshort` 51
`\ifglshassymbol` 51
`\ifglstranslate` 21
`\ifglshasused` 49, 76
`\ifglshasxindy` 23
`index (option)` 26
`index (style)` 282
`indexgroup (style)` 283
`indexhypergroup (style)` 283
`indexonlyfirst (option)` 22
`inline (style)` 248
`\inputencodingname` 24
`\istfile` 160
`\istfilename` 32
`\item` 192, 251, 282, 283

L

`link text` 78
`list (style)` 251
`listdotted (style)` 253
`listgroup (style)` 251
`listhypergroup (style)` 252
`\loadglsentries` 12, 78
`long (key)` 59
`long (style)` 254
`long-sc-short (acrstyle)` 204
`long-sc-short-desc (acrstyle)` ..
..... 206, 332
`long-short (acrstyle)` 203, 330
`long-short-desc (acrstyle)` . 205, 332
`long-sm-short (acrstyle)` 205
`long-sm-short-desc (acrstyle)` ..
..... 206, 333
`long3col (style)` 256
`long3colborder (style)` 256
`long3colheader (style)` 256
`long3colheaderborder (style)` .. 257
`long4col (style)` 257
`long4colborder (style)` 258

`long4colheader (style)` 258
`long4colheaderborder (style)` .. 258
`longaccess (key)` 312
`longborder (style)` 255
`longheader (style)` 255
`longheaderborder (style)` 255
`\longnewglossaryentry` 56, 67
`longplural (key)` 59
`longpluralaccess (key)` 312
`\longprovideglossaryentry` ... 68
`longragged (style)` 260
`longragged3col (style)` 262
`longragged3colborder (style)` .. 262
`longragged3colheader (style)` .. 263
`longragged3colheaderborder`
`(style)` 263
`longraggedborder (style)` 261
`longraggedheader (style)` 261
`longraggedheaderborder (style)` 261
`longtable (environment)`
..... 8, 231, 254–265
`longtable package` 254, 260

M

`\makefirsttuc` 244
`makeglossaries` 23, 32, 46, 53, 155, 171
`\makeglossaries`
..... 28, 32, 33, 53, 55, 154, 156
`\makeglossary` 156
`makeindex` 359
`makeindex` 10, 23, 29, 32–34,
53, 55, 56, 75, 92, 96, 146, 149,
152, 153, 164, 165, 188, 189, 291
`delim_n` 34
`delim_r` 34
`page_compositor` 32
`special characters` 94, 95, 146
`makeindex (option)` 23
`\makenoidxglossaries` 20, 156
`mcolalmtree (style)` 268
`mcolalmtreegroup (style)` 268
`mcolalmtreehypergroup (style)` . 269
`mcolindex (style)` 266
`mcolindexgroup (style)` 266
`mcolindexhypergroup (style)` ... 266
`mcoltree (style)` 266
`mcoltreegroup (style)` 267
`mcoltreehypergroup (style)` 267
`mcoltreename (style)` 267

mcoltreenonamegroup (style) ... 267
 mcoltreenonamehypergroup
 (style) 268
 memoir class 161
 mfirstuc package 1
 \mfirstucMakeUppercase 245
 multicol package 265
 multicols (environment) 266

N

name (key) 56
 \new@glossaryentry 63
 \newacronym .. 22, 23, 59, 78, 195, 196
 \newacronymhook 200
 \newacronymstyle 202
 \newglossary 15, 53, 55, 153, 155, 181
 \newglossaryentry 56, 62, 78, 195, 196
 \newglossaryentry options
 access 312, 314
 counter 57
 description 22, 55, 56,
 60, 62, 68, 117, 139, 196, 223, 311
 descriptionaccess 314, 316
 descriptionplural 118, 311
 descriptionpluralaccess .. 314, 316
 first 57, 71,
 101, 112, 140, 221, 226, 227, 311
 firstaccess 314, 316
 firstplural 57, 114, 140, 311
 firstpluralaccess 314, 316
 format 148
 long 85, 143, 312
 longaccess 315, 317
 longplural 143, 312
 longpluralaccess 315, 317
 name 55,
 56, 59, 62, 68, 115, 138, 167, 310
 nonnumberlist 58
 parent 58, 62
 plural 56, 71, 113, 311
 pluralaccess 314, 316
 prefix 238
 prefixfirst 238
 prefixfirstplural 238
 prefixplural 238
 see 7, 58, 155, 156
 short 85, 142, 311
 shortaccess 315, 317
 shortplural 143, 311

shortpluralaccess 315, 317
 sort 56, 141, 188, 189
 symbol 56, 57, 119, 217,
 219, 221, 226, 257, 273, 310–312
 symbolaccess 314, 316
 symbolplural 120, 311
 symbolpluralaccess 314, 316
 text 56,
 57, 101, 111, 139, 217, 221, 311
 textaccess 314, 316
 type 12, 57, 78, 141
 user1 121, 141, 312
 user2 122, 141
 user3 123, 141
 user4 124, 142
 user5 125, 142
 user6 126, 142, 312

\newglossarystyle 191
 nogroupskip (option) 9
 nohypertypes (option) 16
 \noist 153, 237, 296
 nolist (option) 8
 nolong (option) 8
 nomain (option) 13
 nonumberlist (key) 58
 nonumberlist (option) 7
 \nopostdesc 31
 nopostdot (option) 9
 noredefwarn (option) 16
 nostyles (option) 8
 nosuper (option) 8
 notranslate (option) 21
 notree (option) 8
 nowarn (option) 16
 numberedsection (option) 6
 numberline (option) 5
 numbers (option) 25

O

\oldacronym 195
 order (option) 23

P

package options:
 acronym 13, 29, 168, 196
 true 14
 acronym 13
 acronymlists 15
 acronyms 14
 compatible-2.07 25

compatible-3.07	25	savenumberlist	7
counter	15	savewrites	24, 364
counter	15	false	154
description	221, 222	true	154, 160
description	22	savewrites	24
dua	219, 221, 222	section	6, 36
dua	23	section	6
entrycounter	183	seeautonumberlist	7
true	9	shotcuts	23
entrycounter	9	smallcaps	23
entrycounterwithin	9	smaller	23
footnote	102, 104–	sort	
106, 108–110, 217, 219, 221, 223		def	10
footnote	22	standard	10
hyperfirst		use	10
false	102, 104–106, 108–110	sort	10
hyperfirst	22	style	7, 231
index	26	style	7
indexonlyfirst	366	subentrycounter	183
indexonlyfirst	22	subentrycounter	9
makeindex	150, 237	symbols	25
makeindex	23	toc	5
nogroupskip	9	true	5
nohypertypes	16	toc	5
nolist	231	translate	21
nolist	8	false	21
nolong	231, 254	translate	21
nolong	8	translator	21
nomain	12, 13	ucmark	9
nomain	13	xindy	24, 150, 237
nonumberlist	7	xindy	24
nonumberlist	7	xindygloss	24
nopostdot	9	xindynoglsnumbers	24
noredefwarn	16	\pagelistname	29
nostyles	8	parent (key)	58
nosuper	231	\PGLS	242
nosuper	8	\Pgls	241
notranslate	21	\pgls	240
notree	231	\PGLSpl	243
notree	8	\Pglspl	242
nowarn	16	\pglspl	240
numberedsection	6	\phantomsection	35–37
numberline	5	plural (key)	56
numberline	5	pluralaccess (key)	311
numbers	25	polyglossia package	27, 30
order	23	\printacronyms	13
sanitize	19, 56, 138, 139	\PrintChanges	5
sanitize	21	\printglossaries	
sanitizesort	20 12, 34, 53, 55, 156, 168, 246	

<code>\printglossary</code>	34, 35, 53, 156, 168, 170, 181, 246
<code>\printglossary options</code>	
<code>nogroupskip</code>	182
<code>nonumberlist</code>	182
<code>numberedsection</code>	181
<code>style</code>	181
<code>title</code>	181
<code>toctitle</code>	181
<code>type</code>	12, 167, 181
<code>\printnoidxglossaries</code>	169
<code>\printnoidxglossary</code>	168, 181
<code>\printnoidxglossary options</code>	
<code>sort</code>	182
<code>\provideglossaryentry</code>	62
R	
<code>\renewacronymstyle</code>	203
<code>\renewglossarystyle</code>	192
<code>\roman</code>	42
S	
<code>sanitize (option)</code>	21
<code>sanitizesort (option)</code>	20
<code>savenumberlist (option)</code>	7
<code>savewrites (option)</code>	24
<code>sc-short-long (acrstyle)</code>	205
<code>sc-short-long-desc (acrstyle)</code> ..	207, 333
<code>\scantokens</code>	48
<code>\section</code>	48
<code>section (option)</code>	6
<code>see (key)</code>	58
<code>seeautonumberlist (option)</code>	7
<code>\seename</code>	30
<code>\SetAcronymLists</code>	15
<code>\SetAcronymStyle</code>	14, 229
<code>\setacronymstyle</code>	202
<code>\SetCustomDisplayStyle</code>	229
<code>\SetCustomStyle</code>	230
<code>\SetDefaultAcronymDisplayStyle</code>	214
<code>\SetDefaultAcronymStyle</code>	215
<code>\SetDescriptionAcronymDisplayStyle</code>	219
<code>\SetDescriptionAcronymStyle</code> ..	221
<code>\SetDescriptionDUAAcronymDisplayStyle</code>	218
<code>\SetDescriptionDUAAcronymStyle</code>	219
<code>\SetDescriptionFootnoteAcronymDisplayStyle</code>	216
<code>\SetDescriptionFootnoteAcronymStyle</code>	217
<code>\SetDUADisplayStyle</code>	227
<code>\SetDUASStyle</code>	228
<code>\setentrycounter</code>	190
<code>\SetFootnoteAcronymDisplayStyle</code>	222
<code>\SetFootnoteAcronymStyle</code> ...	223
<code>\SetGenericNewAcronym</code>	201
<code>\setglossarypreamble</code>	35
<code>\setglossarysection</code>	36
<code>\setglossarystyle</code>	191
<code>\setglossentrycompatibility</code>	188
<code>\SetSmallAcronymDisplayStyle</code>	224
<code>\SetSmallAcronymStyle</code>	226
<code>\setstylefile</code>	32
<code>\setupglossaries</code>	27
<code>short (key)</code>	59
<code>short-long (acrstyle)</code>	204, 331
<code>short-long-desc (acrstyle)</code> .	206, 333
<code>shortaccess (key)</code>	311
<code>shortplural (key)</code>	59
<code>shortpluralaccess (key)</code>	311
<code>shotcuts (option)</code>	23
<code>\showacronymlists</code>	236
<code>\showglocounter</code>	233
<code>\showglodesc</code>	234
<code>\showglodescaccess</code>	346
<code>\showglodescplural</code>	234
<code>\showglodescpluralaccess</code> ...	346
<code>\showglofirst</code>	232
<code>\showglofirstaccess</code>	345
<code>\showglofirstpl</code>	232
<code>\showglofirstpluralaccess</code> ..	345
<code>\showgloflag</code>	235
<code>\showgloindex</code>	235
<code>\showglolevel</code>	232
<code>\showgloloclist</code>	235
<code>\showglolong</code>	235
<code>\showglolongaccess</code>	346
<code>\showglolongpluralaccess</code> ...	346
<code>\showgloname</code>	234
<code>\showglonameaccess</code>	345
<code>\showgloparent</code>	231
<code>\showgloplural</code>	232
<code>\showglopluralaccess</code>	345
<code>\showgloshort</code>	235

`\showgloshortaccess` 346
`\showgloshortpluralaccess` .. 346
`\showglosort` 234
`\showglossaries` 236
`\showglossarycounter` 236
`\showglossaryentries` 237
`\showglossaryin` 236
`\showglossaryout` 236
`\showglossarytitle` 236
`\showglosymbol` 234
`\showglosymbolaccess` 345
`\showglosymbolplural` 235
`\showglosymbolpluralaccess` . 345
`\showglotext` 232
`\showglotextaccess` 345
`\showglotype` 232
`\showglouser` 233
`\showglouserii` 233
`\showglouseriii` 233
`\showglouseriv` 233
`\showglouserv` 233
`\showglouservi` 234
`sm-short-long` (acrstyle) 205
`sm-short-long-desc` (acrstyle) ..
..... 207, 334
`smallcaps` (option) 23
`smaller` (option) 23
`\SmallNewAcronymDef` 225, 343
`sort` (key) 56
`sort` (option) 10
`style` (option) 7
`subentrycounter` (option) 9
`\subglossentry` 186
`\subitem` 283
`sublistdotted` (style) 253
`\subsubitem` 283
`super` (style) 269
`super3col` (style) 271
`super3colborder` (style) 272
`super3colheader` (style) 272
`super3colheaderborder` (style) . 272
`super4col` (style) 273
`super4colborder` (style) 274
`super4colheader` (style) 274
`super4colheaderborder` (style) . 274
`superborder` (style) 270
`superheader` (style) 270
`superheaderborder` (style) 271
`superragged` (style) 276

`superragged3col` (style) 278
`superragged3colborder` (style) . 279
`superragged3colheader` (style) . 279
`superraggedborder` (style) 277
`superraggedheader` (style) 277
`superraggedheaderborder` (style) 278
`superraggedright3colheaderborder`
(style) 280
`supertabular` (environment) ...
..... 8, 231, 269–282
`supertabular` package .. 8, 231, 269, 276
`symbol` (key) 57
`symbolaccess` (key) 311
`\symbolname` 29
`symbolplural` (key) 57
`symbolpluralaccess` (key) 311
`symbols` (option) 25

T

`text` (key) 56
`textaccess` (key) 311
`textcase` package 3
`\theequation` 165
`theglossary` (environment)
..... 5, 16, 34, 35,
185, 186, 191, 267, 268, 284–287
`\theHequation` 165
`theindex` (environment) 282
`toc` (option) 5
`\translate` 30
`translate` (option) 21
`translator` package 12–
14, 25–27, 30, 168, 346, 355–359
`tree` (style) 284
`treegroup` (style) 284
`treehypergroup` (style) 285
`treenoname` (style) 285
`treenonamegroup` (style) 286
`treenonamehypergroup` (style) .. 286
`type` (key) 57

U

`ucmark` (option) 9
`user1` (key) 58
`user2` (key) 58
`user3` (key) 59
`user4` (key) 59
`user5` (key) 59
`user6` (key) 59

W	
\warn@nomakeglossaries	154
\warn@noprintglossary	168
\writeist	32, 39, 41, 44, 147, 290, 292
X	
\xcapitalisewords	246
\xglaccsupp	315
xindy	359
xindy	10, 23, 24, 32, 33, 38, 41, 43, 45–47, 75, 99, 147–149, 164, 171, 188, 237, 291
xindy (option)	24
xindygloss (option)	24
xindynoglsnumbers (option)	24
\xmakefirstuc	245
xspace package	4, 195