

# Documented Code For glossaries v4.05

Nicola L.C. Talbot

Dickimaw Books

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

2014-03-11

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 v4.05:  $\TeX$ 2e Package to Assist Generating Glossaries”.

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

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

**INSTALL** Installation instructions.

**CHANGES** Change log.

**README** Package summary.

# Contents

<b>1 Main Package Code</b>	<b>3</b>
1.1 Package Definition	3
1.2 Package Options	5
1.3 Default values	28
1.4 Xindy	38
1.5 Loops and conditionals	47
1.6 Defining new glossaries	52
1.7 Defining new entries	55
1.8 Resetting and unsetting entry flags	75
1.9 Loading files containing glossary entries	77
1.10 Using glossary entries in the text	78
1.10.1 Links to glossary entries	88
1.10.2 Displaying entry details without adding information to the glossary	134
1.11 Adding an entry to the glossary without generating text	141
1.12 Creating associated files	142
1.13 Writing information to associated files	156
1.14 Glossary Entry Cross-References	161
1.15 Displaying the glossary	163
1.16 Acronyms	191
1.17 Predefined acronym styles	196
1.18 Predefined Glossary Styles	227
1.19 Debugging Commands	227
1.20 Compatibility with version 2.07 and below	233
<b>2 Prefix Support (glossaries-prefix Code)</b>	<b>233</b>
<b>3 Mfirstuc Documented Code</b>	<b>240</b>
<b>4 Glossary Styles</b>	<b>242</b>
4.1 Glossary hyper-navigation definitions (glossary-hypernav package)	242
4.2 In-line Style (glossary-inline.sty)	244
4.3 List Style (glossary-list.sty)	247
4.4 Glossary Styles using longtable (the glossary-long package)	250
4.5 Glossary Styles using longtable (the glossary-longragged package)	256
4.6 Glossary Styles using multicol (glossary-mcols.sty)	261
4.7 Glossary Styles using supertabular environment (glossary-super package)	265
4.8 Glossary Styles using supertabular environment (glossary-superragged package)	272
4.9 Tree Styles (glossary-tree.sty)	278
<b>5 glossaries-compatible-207</b>	<b>285</b>

<b>6 Accessibility Support (glossaries-accsupp Code)</b>	<b>305</b>
6.1 Defining Replacement Text . . . . .	306
6.2 Accessing Replacement Text . . . . .	310
6.3 Displaying the Glossary . . . . .	325
6.4 Acronyms . . . . .	326
6.5 Debugging Commands . . . . .	341
<b>7 Multi-Lingual Support</b>	<b>342</b>
7.1 Babel Captions . . . . .	342
7.2 Polyglossia Captions . . . . .	348
7.3 Brazilian Dictionary . . . . .	351
7.4 Danish Dictionary . . . . .	352
7.5 Dutch Dictionary . . . . .	352
7.6 English Dictionary . . . . .	352
7.7 French Dictionary . . . . .	353
7.8 German Dictionary . . . . .	353
7.9 Irish Dictionary . . . . .	353
7.10 Italian Dictionary . . . . .	353
7.11 Magyar Dictionary . . . . .	354
7.12 Polish Dictionary . . . . .	354
7.13 Serbian Dictionary . . . . .	354
7.14 Spanish Dictionary . . . . .	355
<b>Glossary</b>	<b>355</b>
<b>Change History</b>	<b>355</b>
<b>Index</b>	<b>375</b>

# 1 Main Package Code

## 1.1 Package Definition

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

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries}[2014/03/11 v4.05 (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
```

```
225 }
```

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

`\glsdefaulttype`

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

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

`\acronymtype`

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

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

```
228 \@gls@declareoption{nomain}{%
229   \let\glsdefaulttype\relax
230   \renewcommand*{\glsdefmain}{}}%
231 }
```

`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.

```
232 \define@boolkey{glossaries.sty}{gls}{acronym}[true]{%
233   \ifglsacronym
234     \renewcommand*{\@gls@do@acronymsdef}{%
235       \DeclareAcronymList{acronym}%
236       \newglossary[alg]{acronym}{acr}{acn}{\acronymname}%
237       \renewcommand*{\acronymtype}{acronym}%
238     }%
239   \else
240     \let\@gls@do@acronymsdef\relax
241   \fi
242 }
```

`\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.

```
243 \AtBeginDocument{%
244   \ifglsacronym
245     \ifbool{glscompatible-3.07}%
246     {%
247     {%
248       \providecommand*{\printacronyms}[1][1]{%
249         \printglossary[type=\acronymtype,#1]}%
250     }%
251   \fi
252 }
```

`@gls@do@acronymsdef` Set default value

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

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

```
254 \@gls@declareoption{acronyms}{%
255   \glsacronymtrue
256   \renewcommand*{\@gls@do@acronymsdef}{%
257     \DeclareAcronymList{acronym}%
258     \newglossary[alg]{acronym}{acr}{acn}{\acronymname}%
259     \renewcommand*{\acronymtype}{acronym}%
260   }
```

```

260 }%
261 }

```

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

```

262 \newcommand*{\@glsacronymlists}{ }

```

`\@addtoacronymlists`

```

263 \newcommand*{\@addtoacronymlists}[1]{%
264   \ifx\@glsacronymlists\@empty
265     \protected@xdef\@glsacronymlists{#1}%
266   \else
267     \protected@xdef\@glsacronymlists{\@glsacronymlists,#1}%
268   \fi
269 }

```

`\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.)

```

270 \newcommand*{\DeclareAcronymList}[1]{%
271   \glsIfListOfAcronyms{#1}{ }\@addtoacronymlists{#1}}%
272 }

```

`\glsIfListOfAcronyms`

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

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

```

273 \newcommand{\glsIfListOfAcronyms}[1]{%
274   \edef\@do@gls@islistofacronyms{%
275     \noexpand\@gls@islistofacronyms{#1}{\@glsacronymlists}}%
276   \@do@gls@islistofacronyms
277 }

```

Internal command requires label and list to be expanded:

```

278 \newcommand{\@gls@islistofacronyms}[4]{%
279   \def\@gls@islistofacronyms##1,#1,##2\end@gls@islistofacronyms{%
280     \def\@before{##1}\def\@after{##2}}%
281   \gls@islistofacronyms,#2,#1,\@nil\end@gls@islistofacronyms
282   \ifx\@after\@nnil

```

Not found

```

283   #4%
284   \else

```

Found

```

285   #3%
286   \fi
287 }

```

`if@glsisacronymlist` Convenient boolean.

```

288 \newif\if@glsisacronymlist

```

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

```

289 \newcommand*\gls@checkisacronymlist}[1]{%
290   \glsIfListOfAcronyms{#1}%
291   {\@glsisacronymlisttrue}{\@glsisacronymlistfalse}%
292 }

```

`\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.)

```

293 \newcommand*\SetAcronymLists}[1]{%
294   \renewcommand*\@glsacronymlists{#1}%
295 }

```

`acronymlists`

```

296 \define@key{glossaries.sty}{acronymlists}{%
297   \DeclareAcronymList{#1}%
298 }

```

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`

```

299 \newcommand{\glscounter}{page}

```

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

```

300 \define@key{glossaries.sty}{counter}{%
301   \renewcommand*\@glscounter{#1}%
302 }

```

`\@gls@nohyperlist`

```

303 \newcommand*\@gls@nohyperlist{}

```

`sDeclareNoHyperList`

```

304 \newcommand*\GlsDeclareNoHyperList}[1]{%
305   \ifdefempty\@gls@nohyperlist
306   {%
307     \renewcommand*\@gls@nohyperlist{#1}%
308   }%
309   {%
310     \appto\@gls@nohyperlist{,#1}%
311   }%
312 }

```

nohypertypes

```
313 \define@key{glossaries.sty}{nohypertypes}{%
314   \GlsDeclareNoHyperList{#1}%
315 }
```

\GlossariesWarning Prints a warning message.

```
316 \newcommand*{\GlossariesWarning}[1]{%
317   \PackageWarning{glossaries}{#1}%
318 }
```

sariesWarningNoLine Prints a warning message without the line number.

```
319 \newcommand*{\GlossariesWarningNoLine}[1]{%
320   \PackageWarningNoLine{glossaries}{#1}%
321 }
```

nowarn Define package option to suppress warnings

```
322 \@gls@declareoption{nowarn}{%
323   \renewcommand*{\GlossariesWarning}[1]{}%
324   \renewcommand*{\GlossariesWarningNoLine}[1]{}%
325 }
```

@warnonglossdefined Issue a warning if overriding \printglossary

```
326 \newcommand*{\@gls@warnonglossdefined}{%
327   \GlossariesWarning{Overriding \string\printglossary}%
328 }
```

rnontheglossdefined Issue a warning if overriding theglossary

```
329 \newcommand*{\@gls@warnontheglossdefined}{%
330   \GlossariesWarning{Overriding ‘theglossary’ environment}%
331 }
```

noredefwarn Suppress warning on redefinition of \printglossary

```
332 \@gls@declareoption{noredefwarn}{%
333   \renewcommand*{\@gls@warnonglossdefined}{}%
334   \renewcommand*{\@gls@warnontheglossdefined}{}%
335 }
```

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

```
336 \newcommand*{\@gls@sanitizedesc}{%
337 }
338 %\end{macro}
339 %
340 %\begin{macro}{\glssetexpandfield}
```



```

341%\changes{3.13a}{2013-11-05}{new}
342%\begin{definition}
343%\cs{glssetexpandfield}\marg{field}
344%\end{definition}
345% Sets field to always expand.
346%   \begin{macrocode}
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 }

lsnoidxstripaccents
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   \glsssetexpandfield{name}%
448 \fi
449 }

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

```

#### sanitizesort

```

460 \define@boolkey{glossaries.sty}[glS]{sanitizesort}[true]{%
461   \ifglsssanitizesort
462     \glsssetnoexpandfield{sortvalue}%
463     \renewcommand*{\@glS@noidx@setsanitizesort}{%
464       \glsssanitizesorttrue
465       \glsssetnoexpandfield{sortvalue}%
466     }%
467   \else
468     \glsssetexpandfield{sortvalue}%
469     \renewcommand*{\@glS@noidx@setsanitizesort}{%
470       \glsssanitizesortfalse
471       \glsssetexpandfield{sortvalue}%
472     }%
473   \fi
474 }

```

Default setting:

```

475 \glsssanitizesorttrue
476 \glsssetnoexpandfield{sortvalue}%

```

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

```

477 \newcommand*{\@glS@noidx@setsanitizesort}{%
478   \glsssanitizesortfalse
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@nottranslatorhook{\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 glos-
saries:
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   \glSxindy@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{glScompatible-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]}%
613 }%
```

numbers    Create a “symbols” glossary type

```
614 \@gls@declareoption{numbers}{%
615   \let\@gls@do@numbersdef\@gls@numbersdef
616 }
```

Default is not to define the numbers glossary:

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

\@gls@numbersdef

```
618 \newcommand*{\@gls@numbersdef}{%
619   \newglossary[nlg]{numbers}{nls}{nlo}{\glsnumbersgroupname}%
620   \newcommand*{\printnumbers}[1][]{\printglossary[type=numbers,##1]}%
621 }%
```

index    Create an “index” glossary type

```
622 \@gls@declareoption{index}{%
623   \let\@gls@do@indexdef\@gls@indexdef
624 }
```

Default is not to define index glossary:

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

`\@gls@indexdef`

```

626 \newcommand*{\@gls@indexdef}{%
627   \newglossary[ilg]{index}{ind}{idx}{\indexname}%
628   \newcommand*{\printindex}[1] [] {\printglossary[type=index,##1]}%
629   \newcommand*{\newterm}[2] [] {%
630     \newglossaryentry{##2}%
631     {type={index},name={##2},description={\nopostdesc},##1}}
632 }%

```

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

```

633 \@for\CurrentOption := \@declaredoptions\do{%
634   \ifx\CurrentOption\@empty
635   \else
636     \@expandtwoargs
637     \in@ {,\CurrentOption ,}{,\@classoptionslist,\@curroptions,}%
638     \ifin@
639       \@use@option
640       \expandafter \let\csname ds@\CurrentOption\endcsname\@empty
641     \fi
642   \fi
643 }

```

Now process options passed to the package:

```
644 \ProcessOptionsX
```

Load backward compatibility stuff:

```
645 \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:

```

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

```

Now define `\setupglossaries`:

```

649 \newcommand*{\setupglossaries}[1]{%
650   \renewcommand*{\@gls@setacrstyle}{}%
651   \ifglsacrshortcuts
652     \def\@gls@setupshortcuts{\glsacrshortcutstrue}%
653   \else
654     \def\@gls@setupshortcuts{%
655       \ifglsacrshortcuts
656         \DefineAcronymSynonyms
657       \fi
658     }%
659   \fi
660   \glsacrshortcutsfalse
661   \let\@gls@do@numbersdef\relax
662   \let\@gls@do@symbolssdef\relax

```

```

663 \let\@gls@do@indexdef\relax
664 \let\@gls@do@acronymsdef\relax
665 \setkeys{glossaries.sty}{#1}%
666 \@gls@setacrstyle
667 \@gls@setupshortcuts
668 \@gls@do@acronymsdef
669 \@gls@do@numbersdef
670 \@gls@do@symbolssdef
671 \@gls@do@indexdef
672 }

```

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

```

673 \ifglstranslate
674 \ifpackageloaded{polyglossia}%
675 {%
    polyglossia fakes babel so need to check for polyglossia first.
676 }%
677 {%
678     \@ifpackageloaded{babel}%
679     {%
680         \IfFileExists{translator.sty}%
681         {%
682             \RequirePackage{translator}%
683         }%
684     }%
685 }%
686 {}
687 }
688 \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}`).

```

689 \ifthenelse{\equal{\glscounter}{section}}{%
690 {%
691     \ifcsundef{chapter}{}%
692     {%
693         \let\@gls@old@chapter\@chapter
694         \def\@chapter[#1]#2{\@gls@old@chapter[#1]{#2}%
695             \ifcsundef{hyperdef}{\hyperdef{section}{\thesection}}}%
696     }%

```

```
697 }%
698 {}
```

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

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

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

```
700 \newcommand*{\@onlypremakeg}[1]{%
701   \ifx\@gls@onlypremakeg\@empty
702     \def\@gls@onlypremakeg{#1}%
703   \else
704     \expandafter\toks@\expandafter{\@gls@onlypremakeg}%
705     \edef\@gls@onlypremakeg{\the\toks@,\noexpand#1}%
706   \fi
707 }
```

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

```
708 \newcommand*{\@disable@onlypremakeg}{%
709   \@for\@thiscs:=\@gls@onlypremakeg\do{%
710     \expandafter\@disable@premakecs\@thiscs%
711   }}
```

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

```
712 \newcommand*{\@disable@premakecs}[1]{%
713   \def#1{\PackageError{glossaries}{\string#1\space may only be
714     used before \string\makeglossaries}{You can't use
715     \string#1\space after \string\makeglossaries}}%
716 }
```

### 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`

```
717 \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`

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

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

```
719 \newcommand*{\glsettoctitle}[1]{%
720 \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`

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

`\descriptionname`

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

`\symbolname`

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

`\pagelistname`

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

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

`glssymbolsgroupname`

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

`glnumbersgroupname`

```
726 \providecommand*{\glnumbersgroupname}{Numbers}
```

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

```
727 \newcommand*{\glpluralsuffix}{s}
```

`\seename`

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

`\andname`

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

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

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

```
730 \newcommand*{\addglossarytocaptions}[1]{%
731 \ifcsundef{captions#1}{}%
732 {%
733 \expandafter\let\expandafter\@gls@tmp\csname captions#1\endcsname
```

```

734 \expandafter\toks@\expandafter{\@gls@tmp
735 \renewcommand*{\glossaryname}{\translate{Glossary}}}%
736 }%
737 \expandafter\edef\csname captions#1\endcsname{\the\toks@}%
738 }%
739 }

740 \ifglstranslate

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

741 \@ifpackageloaded{translator}{%
742 \usedictionary{glossaries-dictionary}%
743 \addglossarytocaptions{portuges}%
744 \addglossarytocaptions{portuguese}%
745 \addglossarytocaptions{brazil}%
746 \addglossarytocaptions{brazilian}%
747 \addglossarytocaptions{danish}%
748 \addglossarytocaptions{dutch}%
749 \addglossarytocaptions{afrikaans}%
750 \addglossarytocaptions{english}%
751 \addglossarytocaptions{UKenglish}%
752 \addglossarytocaptions{USenglish}%
753 \addglossarytocaptions{american}%
754 \addglossarytocaptions{australian}%
755 \addglossarytocaptions{british}%
756 \addglossarytocaptions{canadian}%
757 \addglossarytocaptions{newzealand}%
758 \addglossarytocaptions{french}%
759 \addglossarytocaptions{frenchb}%
760 \addglossarytocaptions{francais}%
761 \addglossarytocaptions{acadian}%
762 \addglossarytocaptions{canadien}%
763 \addglossarytocaptions{german}%
764 \addglossarytocaptions{germanb}%
765 \addglossarytocaptions{austrian}%
766 \addglossarytocaptions{naustrian}%
767 \addglossarytocaptions{ngerman}%
768 \addglossarytocaptions{irish}%
769 \addglossarytocaptions{italian}%
770 \addglossarytocaptions{magyar}%
771 \addglossarytocaptions{hungarian}%
772 \addglossarytocaptions{polish}%
773 \addglossarytocaptions{spanish}%
774 \renewcommand*{\glstoctitle}[1]{%
775 \ifthenelse{\equal{#1}{main}}{%
776 \translatelet{\glossarytoctitle}{Glossary}}{%
777 \ifthenelse{\equal{#1}{acronym}}{%
778 \translatelet{\glossarytoctitle}{Acronyms}}{%
779 \def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}}%
780 \renewcommand*{\glossaryname}{\translate{Glossary}}%

```

```

781 \renewcommand*{\acronymname}{\translate{Acronyms}}}%
782 \renewcommand*{\entryname}{\translate{Notation (glossaries)}}}%
783 \renewcommand*{\descriptionname}{%
784 \translate{Description (glossaries)}}}%
785 \renewcommand*{\symbolname}{\translate{Symbol (glossaries)}}}%
786 \renewcommand*{\pagelistname}{%
787 \translate{Page List (glossaries)}}}%
788 \renewcommand*{\glssymbolsgroupname}{%
789 \translate{Symbols (glossaries)}}}%
790 \renewcommand*{\glsnumbersgroupname}{%
791 \translate{Numbers (glossaries)}}}%
792 }{%

793 \@ifpackageloaded{polyglossia}%
794 {\RequirePackage{glossaries-polyglossia}}}%
795 {%
796 \@ifpackageloaded{babel}{%
797 \RequirePackage{glossaries-babel}}}%
798 }}
799 \else

800 \@gls@notranslatorhook
801 \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.

```
802 \DeclareRobustCommand*{\nopostdesc}{}
```

`\@nopostdesc` Suppress next description terminator.

```

803 \newcommand*{\@nopostdesc}{%
804 \let\org@glspostdescription\glspostdescription
805 \def\glspostdescription{%
806 \let\glspostdescription\org@glspostdescription}%
807 }

```

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

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

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

```
809 \newcommand{\glspar}{\par}
```

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

```

810 \ifglxindy
811 \newcommand{\setStyleFile}[1]{%
812 \renewcommand{\istfilename}{#1.xdy}}
813 \else
814 \newcommand{\setStyleFile}[1]{%
815 \renewcommand{\istfilename}{#1.ist}}
816 \fi

```

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

```
817 \@onlypremakeg\setStyleFile
```

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

```
\istfilename
```

```
818 \ifglxindy
819   \def\istfilename{\jobname.xdy}
820 \else
821   \def\istfilename{\jobname.ist}
822 \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 it is not required by  $\TeX$ , `\@istfilename` ignores its argument.

```
\@istfilename
```

```
823 \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
```

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

```
\glsSetCompositor
```

 Sets the compositor.

```
825 \newcommand*{\glsSetCompositor}[1]{%
826   \renewcommand*{\glscompositor}{#1}}
```

Only use before `\makeglossaries`

```
827 \@onlypremakeg\glsSetCompositor
```

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

```
@glsAlphacompositor
```

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

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



`\glsSetAlphaCompositor` Sets the alpha compositor.

```
829 \ifglxsindy
830   \newcommand*\glsSetAlphaCompositor[1]{%
831     \renewcommand*\@glsAlphacompositor{#1}}
832 \else
833   \newcommand*\glsSetAlphaCompositor[1]{%
834     \glsnxindywarning\glsSetAlphaCompositor}
835 \fi
```

Can only be used before `\makeglossaries`

```
836 \@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.

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

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

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

Only has an effect when used before `\makeglossaries`

```
840 \@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.

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

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

```
842 \newcommand*\{\glsSetSuffixFF}[1]{%
843   \renewcommand*\{\gls@suffixFF}{#1}%
844 }
```

`\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”.

```
845 \ifcsundef{hyperlink}%
846 {%
847   \newcommand*\{\glsnumberformat}[1]{#1}%
848 }%
849 {%
850   \newcommand*\{\glsnumberformat}[1]{\glshypernumber{#1}}%
851 }
```

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`

```
852 \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`

```
853 \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`

```
854 \newcommand*{\glossarypreamble}{%
855   \csuse{@glossarypreamble@currentglossary}%
856 }
```

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

Code provided by Michael Pock.

```
857 \newcommand{\setglossarypreamble}[2][\glsdefaulttype]{%
858   \ifglossaryexists{#1}{%
859     \csgdef{@glossarypreamble@#1}{#2}%
860   }{%
861     \GlossariesWarning{%
862       Glossary ‘#1’ is not defined%
863     }%
864   }%
865 }
```

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

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

```

\glossarypostamble
866 \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.
867 \newcommand*{\glossarysection}[2][\@gls@title]{%
868   \def\@gls@title{#2}%
869   \ifcsundef{phantomsection}%
870     {%
871       \@glossarysection{#1}{#2}%
872     }%
873   {%
874     \p@glossarysection{#1}{#2}%
875   }%

876   \glsglossarymark{\glossarytoctitle}%
877 }

\glsglossarymark Sets the header mark for the glossary. Takes the glossary short (TOC) title as the
                  argument.
878 \ifcsundef{glossarymark}%
879 {%
880   \newcommand{\glsglossarymark}[1]{\glossarymark{#1}}
881 }%
882 {%
883   \@ifclassloaded{memoir}
884   {%
885     \newcommand{\glsglossarymark}[1]{%
886       \ifglsucmark
887         \markboth{\memUHead{#1}}{\memUHead{#1}}%
888       \else
889         \markboth{#1}{#1}%
890       \fi
891     }
892   }%
893   {%
894     \newcommand{\glsglossarymark}[1]{%
895       \ifglsucmark
896         \mkboth{\mfirstucMakeUppercase{#1}}{\mfirstucMakeUppercase{#1}}%
897       \else
898         \mkboth{#1}{#1}%
899       \fi
900     }
901   }
902 }

\glossarymark Provided for backward compatibility:

```

```

903 \providecommand{\glossarymark}[1]{%
904   \ifglsucmark
905     \@mkboth{\mfirstucMakeUppercase{#1}}{\mfirstucMakeUppercase{#1}}%
906   \else
907     \@mkboth{#1}{#1}%
908   \fi
909 }

```

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`

```

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

```

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

`\@glossarysection`

```

912 \newcommand*\@glossarysection[2]{%
913   \ifdefempty\@glossarysecstar
914   {%
915     \csname\@glossarysec\endcsname{#2}%
916   }%
917   {%
918     \csname\@glossarysec\endcsname*{#2}%
919     \@gls@toc{#1}{\@glossarysec}%
920   }%

```

Do automatic labelling if required

```

921   \@glossaryseclabel
922 }

```

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

`\@pglossarysection`

```

923 \newcommand*\@pglossarysection[2]{%
924   \glsclearpage
925   \phantomsection
926   \ifdefempty\@glossarysecstar
927   {%
928     \csname\@glossarysec\endcsname{#2}%
929   }%
930   {%

```

```

931 \gls@toc{#1}{\@glossarysec}%
932 \csname\@glossarysec\endcsname*{#2}%
933 }%
    Do automatic labelling if required
934 \@glossaryseclabel
935 }

```

`\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.

```

936 \newcommand*\gls@doclearpage{%
937 \ifthenelse{\equal{\@glossarysec}{chapter}}%
938 {%
939 \ifcsundef{cleardoublepage}%
940 {%
941 \clearpage
942 }%
943 {%
944 \ifcsdef{if@openright}%
945 {%
946 \if@openright
947 \cleardoublepage
948 \else
949 \clearpage
950 \fi
951 }%
952 {%
953 \cleardoublepage
954 }%
955 }%
956 }%
957 }%
958 }

```

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

```

959 \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`

```

960 \newcommand*\@gls@toc[2]{%
961 \ifglstoc
962 \ifglslnumberline
963 \addcontentsline{toc}{#2}{\numberline{#1}}%

```

```

964 \else
965 \addcontentsline{toc}{#2}{#1}%
966 \fi
967 \fi
968 }

```

## 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

```

969 \newcommand*{\glsnoxindywarning}[1]{%
970 \GlossariesWarning{Not in xindy mode --- ignoring \string#1}%
971 }

```

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

```

972 \ifglsxindy
973 \edef\@xdyattributes{\string"default\string"%
974 \fi

```

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

```

975 \ifglsxindy
976 \edef\@xdyattributelist{}%
977 \fi

```

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

```

978 \ifglsxindy
979 \def\@xdylocref{}
980 \fi

```

`\@gls@ifinlist`

```

981 \newcommand*{\@gls@ifinlist}[4]{%
982 \def\@do@ifinlist##1,#1,##2\end@do@ifinlist{%
983 \def\@gls@listsuffix{##2}%
984 \ifx\@gls@listsuffix\@empty
985 #4%
986 \else
987 #3%
988 \fi
989 }%
990 \@do@ifinlist,#2,#1,\end@do@ifinlist
991 }

```

`\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.

```

992 \ifglxsindy
993   \newcommand*{\@xdycounters}{\glscounter}
994   \newcommand*\GlsAddXdyCounters[1]{%
995     \@for\@gls@ctr:=#1\do{%
      Check if already in list before adding.
996       \edef\@do@addcounter{%
997         \noexpand\@gls@ifinlist{\@gls@ctr}{\@xdycounters}{}%
998         {%
999           \noexpand\edef\noexpand\@xdycounters{\@xdycounters,%
1000             \noexpand\@gls@ctr}%
1001         }%
1002       }%
1003       \@do@addcounter
1004     }
1005   }

Only has an effect before \writeist:
1006   \@onlypremakeg\GlsAddXdyCounters
1007 \else
1008   \newcommand*\GlsAddXdyCounters[1]{%
1009     \glsnoxindywarning\GlsAddXdyAttribute
1010   }
1011 \fi

```

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

```

1012 \newcommand*\@disabled@glssaddxdycounters{%
1013   \PackageError{glossaries}{\string\GlsAddXdyCounters\space
1014     can't be used after \string\GlsAddXdyAttribute}{Move all
1015     occurrences of \string\GlsAddXdyCounters\space before the first
1016     instance of \string\GlsAddXdyAttribute}%
1017 }

```

`\GlsAddXdyAttribute` Adds an attribute.

```

1018 \ifglxsindy

First define internal command that adds an attribute for a given counter (2nd
argument is the counter):
1019   \newcommand*\@glssaddxdyattribute[2]{%

Add to xindy attribute list
1020     \edef\@xdyattributes{\@xdyattributes ^^J \string"#1\string" ^^J
1021       \string"#2#1\string"}%

Add to xindy markup location.
1022     \expandafter\toks@\expandafter{\@xdylocref}%
1023     \edef\@xdylocref{\the\toks@ ^^J%
1024       (markup-locref
1025       :open \string"\string~n%
1026       \expandafter\string\csname glsX#2X#1\endcsname
1027       \string" ^^J

```

```

1028      :close \string"\string" ^^J
1029      :attr \string"#2#1\string"))}%

  Define associated attribute command \glsX<counter>X<attribute>{\Hprefix}{\<n>}
1030      \expandafter\gdef\csname glsX#2X#1\endcsname##1##2{%
1031          \setentrycounter[##1]{#2}\csname #1\endcsname{##2}%
1032      }%
1033  }

  High-level command:
1034  \newcommand*\GlsAddXdyAttribute[1]{%

    Add to comma-separated attribute list
1035      \ifx\@xdyattributelist\empty
1036          \edef\@xdyattributelist{#1}%
1037      \else
1038          \edef\@xdyattributelist{\@xdyattributelist,#1}%
1039      \fi

    Iterate through all specified counters and add counter-dependent attributes:
1040      \@for\@this@counter:=\@xdycounters\do{%
1041          \protected@edef\gls@do@addxdyattribute{%
1042              \noexpand\@glsaddxdyattribute{#1}{\@this@counter}%
1043          }
1044          \gls@do@addxdyattribute
1045      }%

    All occurrences of \GlsAddXdyCounters must be used before this command
1046      \let\GlsAddXdyCounters\@disabled@glsaddxdycounters
1047  }

    Only has an effect before \writeist:
1048  \@onlypremakeg\GlsAddXdyAttribute
1049 \else
1050  \newcommand*\GlsAddXdyAttribute[1]{%
1051      \glsnoxindywarning\GlsAddXdyAttribute}
1052 \fi

```

**redefinedattributes**    Add known attributes for all defined counters

```

1053 \ifglxsindy
1054 \newcommand*\@gls@addpredefinedattributes{%
1055     \GlsAddXdyAttribute{glsnumberformat}
1056     \GlsAddXdyAttribute{textrm}
1057     \GlsAddXdyAttribute{textsf}
1058     \GlsAddXdyAttribute{texttt}
1059     \GlsAddXdyAttribute{textbf}
1060     \GlsAddXdyAttribute{textmd}
1061     \GlsAddXdyAttribute{textit}
1062     \GlsAddXdyAttribute{textup}
1063     \GlsAddXdyAttribute{textsl}
1064     \GlsAddXdyAttribute{textsc}
1065     \GlsAddXdyAttribute{emph}

```



```

1066 \GlsAddXdyAttribute{glshypernumber}
1067 \GlsAddXdyAttribute{hyperrrm}
1068 \GlsAddXdyAttribute{hypersf}
1069 \GlsAddXdyAttribute{hypertt}
1070 \GlsAddXdyAttribute{hyperbf}
1071 \GlsAddXdyAttribute{hypermd}
1072 \GlsAddXdyAttribute{hyperit}
1073 \GlsAddXdyAttribute{hyperup}
1074 \GlsAddXdyAttribute{hypersl}
1075 \GlsAddXdyAttribute{hypersc}
1076 \GlsAddXdyAttribute{hyperemph}
1077 }
1078 \else
1079 \let\@gls@addpredefinedattributes\relax
1080 \fi

```

`\@xdyuseralphabets` List of additional alphabets

```

1081 \def\@xdyuseralphabets{}

```

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

```

1082 \ifglsxindy
1083 \newcommand*{\GlsAddXdyAlphabet}[2]{%
1084 \edef\@xdyuseralphabets{%
1085 \@xdyuseralphabets ^^J
1086 (define-alphabet "#1" (#2))}}
1087 \else
1088 \newcommand*{\GlsAddXdyAlphabet}[2]{%
1089 \glsnoxywarning\GlsAddXdyAlphabet}
1090 \fi

```

This code is only required for xindy:

```

1091 \ifglsxindy

```

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

```

1092 \newcommand*{\@gls@xdy@locationlist}{%
1093 roman-page-numbers,%
1094 Roman-page-numbers,%
1095 arabic-page-numbers,%
1096 alpha-page-numbers,%
1097 Alpha-page-numbers,%
1098 Appendix-page-numbers,%
1099 arabic-section-numbers%
1100 }

```

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.

```

1101 \protected@edef\@gls@roman{\@roman{0}\string"
1102 \string"roman-numbers-lowercase\string" :sep \string"}}%
1103 \@onelevel@sanitize\@gls@roman
1104 \edef\@tmp{\string" \string"roman-numbers-lowercase\string"
1105 :sep \string"}}%
1106 \@onelevel@sanitize\@tmp
1107 \ifx\@tmp\@gls@roman
1108 \expandafter
1109 \edef\csname @gls@xdy@Lclass@roman-page-numbers\endcsname{%
1110 \string"roman-numbers-lowercase\string"%
1111 }%
1112 \else
1113 \expandafter
1114 \edef\csname @gls@xdy@Lclass@roman-page-numbers\endcsname{
1115 :sep \string"\@gls@roman\string"%
1116 }%
1117 \fi

```

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

```

1118 \expandafter\def\csname @gls@xdy@Lclass@Roman-page-numbers\endcsname{%
1119 \string"roman-numbers-uppercase\string"%
1120 }%

```

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

```

1121 \expandafter\def\csname @gls@xdy@Lclass@arabic-page-numbers\endcsname{%
1122 \string"arabic-numbers\string"%
1123 }%

```

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

```

1124 \expandafter\def\csname @gls@xdy@Lclass@alpha-page-numbers\endcsname{%
1125 \string"alpha\string"%
1126 }%

```

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

```

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

```

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

```

1130 \expandafter\def\csname @gls@xdy@Lclass@Appendix-page-numbers\endcsname{%
1131 \string"ALPHA\string"
1132 :sep \string"\@glsAlphacompositor\string"
1133 \string"arabic-numbers\string"%
1134 }

```

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

```

1135 \expandafter\def\csname @gls@xdy@Lclass@arabic-section-numbers\endcsname{%
1136   \string"arabic-numbers\string"
1137   :sep \string"\glscompositor\string"
1138   \string"arabic-numbers\string"%
1139 }%
```

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

```

1140 \def\@xdyuserlocationdefs{}
```

xdyuserlocationnames List of additional user location names

```

1141 \def\@xdyuserlocationnames{}
```

End of xindy-only block:

```

1142 \fi
```

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

```

1143 \ifglsxindy
1144   \newcommand*\GlsAddXdyLocation[3][{}]{%
1145     \def\@gls@tmp{#1}%
1146     \ifx\@gls@tmp\@empty
1147       \edef\@xdyuserlocationdefs{%
1148         \@xdyuserlocationdefs ^^J%
1149         (define-location-class \string"#2\string"^^J\space\space
1150         \space(:sep \string"{}\glsopenbrace\string" #3
1151         :sep \string"\glsclosebrace\string"))
1152       }%
1153     \else
1154       \edef\@xdyuserlocationdefs{%
1155         \@xdyuserlocationdefs ^^J%
1156         (define-location-class \string"#2\string"^^J\space\space
1157         \space(:sep "\glsopenbrace"
1158         #1
1159         :sep "\glsclosebrace\glsopenbrace" #3
1160         :sep "\glsclosebrace"))
1161       }%
1162     \fi
1163     \edef\@xdyuserlocationnames{%
1164       \@xdyuserlocationnames^^J\space\space\space
1165       \string"#1\string"}%
1166   }
```

Only has an effect before `\writeist`:

```

1167 \@onlypremakeg\GlsAddXdyLocation
```

```

1168 \else
1169   \newcommand*\GlsAddXdyLocation}[2]{%
1170     \glsnoxywarning\GlsAddXdyLocation}
1171 \fi

```

ylocationclassorder Define location class order

```

1172 \ifglxindy
1173   \edef\@xdylocationclassorder{^^J\space\space\space
1174     \string"roman-page-numbers\string"^^J\space\space\space
1175     \string"arabic-page-numbers\string"^^J\space\space\space
1176     \string"arabic-section-numbers\string"^^J\space\space\space
1177     \string"alpha-page-numbers\string"^^J\space\space\space
1178     \string"Roman-page-numbers\string"^^J\space\space\space
1179     \string"Alpha-page-numbers\string"^^J\space\space\space
1180     \string"Appendix-page-numbers\string"
1181     \@xdyuserlocationnames^^J\space\space\space
1182     \string"see\string"
1183   }
1184 \fi

```

Change the location order.

yLocationClassOrder

```

1185 \ifglxindy
1186   \newcommand*\GlsSetXdyLocationClassOrder[#1]{%
1187     \def\@xdylocationclassorder{#1}}
1188 \else
1189   \newcommand*\GlsSetXdyLocationClassOrder[#1]{%
1190     \glsnoxywarning\GlsSetXdyLocationClassOrder}
1191 \fi

```

\@xdysortrules Define sort rules

```

1192 \ifglxindy
1193   \def\@xdysortrules{}
1194 \fi

```

\GlsAddSortRule Add a sort rule

```

1195 \ifglxindy
1196   \newcommand*\GlsAddSortRule[#2]{%
1197     \expandafter\toks@\expandafter{\@xdysortrules}%
1198     \protected@edef\@xdysortrules{\the\toks@ ^^J
1199       (sort-rule \string"#1\string" \string"#2\string")}%
1200   }
1201 \else
1202   \newcommand*\GlsAddSortRule[#2]{%
1203     \glsnoxywarning\GlsAddSortRule}
1204 \fi

```

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

```

1205 \ifglxindy
1206   \def\@xdyrequiredstyles{tex}
1207 \fi

```

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

```

1208 \ifglxindy
1209   \newcommand*\GlsAddXdyStyle[1]{%
1210     \edef\@xdyrequiredstyles{\@xdyrequiredstyles,#1}}%
1211 \else
1212   \newcommand*\GlsAddXdyStyle[1]{%
1213     \glsnnoxindywarning\GlsAddXdyStyle}
1214 \fi

```

`\GlsSetXdyStyles` Reset the list of required styles

```

1215 \ifglxindy
1216   \newcommand*\GlsSetXdyStyles[1]{%
1217     \edef\@xdyrequiredstyles{#1}}
1218 \else
1219   \newcommand*\GlsSetXdyStyles[1]{%
1220     \glsnnoxindywarning\GlsSetXdyStyles}
1221 \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.

```

1222 \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.

```

1223 \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.

```

1224 \ifglxindy
1225   \newcommand*\GlsSetXdyLanguage[2][\glsdefaulttype]{%
1226     \ifglossaryexists{#1}{%
1227       \expandafter\def\csname @xdy@#1@language\endcsname{#2}%
1228     }{%
1229       \PackageError{glossaries}{Can't set language type for
1230         glossary type '#1' --- no such glossary}{%

```

```

1231     You have specified a glossary type that doesn't exist}}
1232 \else
1233   \newcommand*\GlsSetXdyLanguage[2][]{%
1234     \glsnoxywarning\GlsSetXdyLanguage}
1235 \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.

```

1236 \def\@gls@codepage#1#2{}

```

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

```

1237 \ifglxindy
1238   \newcommand*\GlsSetXdyCodePage[1]{%
1239     \renewcommand*\@gls@codepage{#1}%
1240   }

```

Suggested by egreg:

```

1241   \AtBeginDocument{%
1242     \ifx\@gls@codepage\@empty
1243       \@ifpackageloaded{fontspec}{\def\@gls@codepage{utf8}}{}%
1244     \fi
1245   }
1246 \else
1247   \newcommand*\GlsSetXdyCodePage[1]{%
1248     \glsnoxywarning\GlsSetXdyCodePage}
1249 \fi

```

`\@xdylettergroups` Store letter group definitions.

```

1250 \ifglxindy
1251   \ifglxindy@glsnumbers
1252     \def\@xdylettergroups{(define-letter-group
1253       \string\glsnumbers\string^^J\space\space\space
1254       :prefixes (\string"0\string" \string"1\string"
1255       \string"2\string" \string"3\string" \string"4\string"
1256       \string"5\string" \string"6\string" \string"7\string"
1257       \string"8\string" \string"9\string")^^J\space\space\space
1258       :before \string"\@glsfirstletter\string")}
1259   \else
1260     \def\@xdylettergroups{}
1261   \fi
1262 \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.

```

1263   \newcommand*\GlsAddLetterGroup[2]{%
1264     \expandafter\toks@\expandafter{\@xdylettergroups}%
1265     \protected@edef\@xdylettergroups{\the\toks@^^J%

```

```

1266     (define-letter-group \string"#1\string"^^J\space\space\space#2))}%
1267 }%

```

## 1.5 Loops and conditionals

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

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

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

```

1268 \newcommand*\forallglossaries[3][\@glo@types]{%
1269   \@for#2:=#1\do{\ifx#2\@empty\else#3\fi}%
1270 }

```

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

```
\forall glossentries[<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.

```

1271 \newcommand*\forallglossentries[3][\glsdefaulttype]{%
1272   \edef\@glo@list{\csname glolist@#1\endcsname}%
1273   \@for#2:=\@glo@list\do
1274   {%
1275     \ifdefempty{#2}{-}{#3}%
1276   }%
1277 }

```

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

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

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

```

1278 \newcommand*\forallglossentries[3][\@glo@types]{%
1279   \expandafter\forallglossaries\expandafter[#1]{\@this@glo@}%
1280   {%
1281     \forallglossentries[\@this@glo@]{#2}{#3}%
1282   }%
1283 }

```

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

```
\if glossary exists{<type>}{<true-text>}{<false-text>}
```

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

```

1284 \newcommand{\ifglossaryexists}[3]{%
1285   \ifcsundef{@glo@#1@out}{#3}{#2}%
1286 }

```

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

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

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

`\glsetoklabel`

```
1287 \newcommand*{\glsetoklabel}[1]{#1}
```

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

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

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

```

1288 \newcommand{\ifglentryexists}[3]{%
1289   \ifcsundef{glo@\glsetoklabel{#1}@name}{#3}{#2}%
1290 }

```

`\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>`.

```

1291 \newcommand*{\ifglused}[3]{%
1292   \ifbool{glo@\glsetoklabel{#1}@flag}{#2}{#3}%
1293 }

```

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

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



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

```

1294 \newcommand{\glsdoifexists}[2]{%
1295   \ifglentryexists{#1}{#2}{%
1296     \PackageError{glossaries}{Glossary entry ‘\glsdetoklabel{#1}’
1297       has not been defined}{You need to define a glossary entry before you
1298       can use it.}}%
1299 }
```

$\backslash\text{glsdoifnoexists}$      $\backslash\text{glsdoifnoexists}\{\langle label \rangle\}\{\langle code \rangle\}$

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

```

1300 \newcommand{\glsdoifnoexists}[2]{%
1301   \ifglentryexists{#1}{%
1302     \PackageError{glossaries}{Glossary entry ‘\glsdetoklabel{#1}’ has already
1303     been defined}{}}{#2}%
1304 }
```

$\backslash\text{glsdoifexistsorwarn}$      $\backslash\text{glsdoifexistsorwarn}\{\langle label \rangle\}\{\langle code \rangle\}$

Generate a warning if entry specified by  $\langle label \rangle$  doesn't exists, otherwise do  $\langle code \rangle$ .

```

1305 \newcommand{\glsdoifexistsorwarn}[2]{%
1306   \ifglentryexists{#1}{#2}{%
1307     \GlossariesWarning{Glossary entry ‘\glsdetoklabel{#1}’
1308       has not been defined}%
1309   }%
1310 }
```

$\backslash\text{ifglshaschildren}$      $\backslash\text{ifglshaschildren}\{\langle label \rangle\}\{\langle true\ part \rangle\}\{\langle false\ part \rangle\}$

```

1311 \newcommand{\ifglshaschildren}[3]{%
1312   \glsdoifexists{#1}%
1313   {%
1314     \def\do@glshaschildren{#3}%
1315     \edef\@gls@thislabel{\glsdetoklabel{#1}}%
1316     \expandafter\forglentries\expandafter
1317     [\csname glo@\@gls@thislabel @type\endcsname]
1318     {\glo@label}%
1319     {%
1320       \letcs@glo@parent{glo@\glo@label @parent}%
1321       \ifdefequal\@gls@thislabel\glo@parent
1322       {%
1323         \def\do@glshaschildren{#2}%
1324         \@endfortrue
1325       }%
1326     }%
1327   }%
```

```

1328     \do@glshaschildren
1329 }%
1330 }

```

```

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

```

```

1331 \newcommand{\ifglshasparent}[3]{%
1332   \glsdoifexists{#1}%
1333   {%
1334     \ifcsempy{glo@glsdetoklabel{#1}@parent}{#3}{#2}%
1335   }%
1336 }

```

```

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

```

```

1337 \newcommand*{\ifglshasdesc}[3]{%
1338   \ifcsempy{glo@glsdetoklabel{#1}@desc}%
1339   {#3}%
1340   {#2}%
1341 }

```

```

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

```

```

1342 \newcommand*{\ifglsdescsuppressed}[3]{%
1343   \ifcsequal{glo@glsdetoklabel{#1}@desc}{@no@post@desc}%
1344   {#2}%
1345   {#3}%
1346 }

```

```

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

```

```

1347 \newcommand*{\ifglshassymbol}[3]{%
1348   \letcs{\@glo@symbol}{glo@glsdetoklabel{#1}@symbol}%
1349   \ifdefempty\@glo@symbol
1350   {#3}%
1351   {%
1352     \ifdefequal\@glo@symbol\@gls@default@value
1353     {#3}%
1354     {#2}%
1355   }%
1356 }

```

```

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

```

```

1357 \newcommand*{\ifglshaslong}[3]{%
1358   \letcs{\@glo@long}{glo@glsdetoklabel{#1}@long}%
1359   \ifdefempty\@glo@long
1360   {#3}%
1361   {%

```

```

1362 \ifdefequal\@glo@long\@gls@default@value
1363 {#3}%
1364 {#2}%
1365 }%
1366 }

```

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

```

1367 \newcommand*\ifglshasshort[3]{%
1368 \letcs{\@glo@short}{glo\@glsdetoklabel{#1}@short}%
1369 \ifdefempty\@glo@short
1370 {#3}%
1371 {%
1372 \ifdefequal\@glo@short\@gls@default@value
1373 {#3}%
1374 {#2}%
1375 }%
1376 }

```

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

```

1377 \newcommand*\ifglshasfield[4]{%
1378 \glsdoifexists{#2}%
1379 {%
1380 \letcs{\@glo@thisvalue}{glo\@glsdetoklabel{#2}@#1}%

```

First check supplied field label is defined.

```

1381 \ifdef\@glo@thisvalue
1382 {%

```

Is defined, so now check if empty.

```

1383 \ifdefempty\@glo@thisvalue
1384 {%

```

Is empty, so doesn't have field set.

```

1385 #4%
1386 }%
1387 {%

```

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

```

1388 \ifdefequal\@glo@thisvalue\@gls@default@value{#4}{#3}%
1389 }%
1390 }%
1391 {%

```

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

```

1392 \@gls@fetchfield{\@gls@thisfield}{#1}%

```

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

```

1393 \ifdef\@gls@thisfield
1394 {%

```

Is defined, so now check if empty.

```
1395      \letcs{\@glo@thisvalue}{glo@glstetoklabel{#2}@\@gls@thisfield}%
1396      \ifdefempty\@glo@thisvalue
1397      {%
```

Is empty so field hasn't been set.

```
1398      #4%
1399      }%
1400      {%
```

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

```
1401      \ifdefequal\@glo@thisvalue\@gls@default@value{#4}{#3}%
1402      }%
1403      }%
1404      {%
```

Not defined.

```
1405      \GlossariesWarning{Unknown entry field '#1'}%
1406      #4%
1407      }%
1408      }%
1409      }%
1410 }
```

## 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

```
1411 \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.

```
1412 \newcommand*\@gls@provide@newglossary{%
1413   \protected@write\@auxout{}\string\providecommand\string\@newglossary[4]{}}%
```

Only need to do this once.

```
1414   \let\@gls@provide@newglossary\relax
1415 }
```

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

```
1416 \newcommand*\defglsentryfmt}[2][\glsdefaulttype]{%
1417   \csgdef{gls@#1@entryfmt}{#2}%
1418 }
```

`\gls@doentryfmt`

```
1419 \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`

```
1420 \newcommand*{\newglossary}[5][glg]{%  
1421 \ifglossaryexists{#2}%  
1422 {%  
1423 \PackageError{glossaries}{Glossary type ‘#2’ already exists}{%  
1424 You can’t define a new glossary called ‘#2’ because it already  
1425 exists}%  
1426 }%  
1427 {%
```

Check if default has been set

```
1428 \ifundef\glsdefaulttype  
1429 {%  
1430 \gdef\glsdefaulttype{#2}%  
1431 }{}}%
```

Add this to the list of glossary types:

```
1432 \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.

```
1433 \expandafter\gdef\csname glolist@#2\endcsname{,}%
```

Store details of this new glossary type:

```
1434 \expandafter\def\csname @glotype@#2@in\endcsname{#3}%  
1435 \expandafter\def\csname @glotype@#2@out\endcsname{#4}%  
1436 \expandafter\def\csname @glotype@#2@title\endcsname{#5}%  
  
1437 \@gls@provide@newglossary  
1438 \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.

```
1439 \ifcsundef{gls@#2@entryfmt}%
1440 {%
1441   \defglsentryfmt[#2]{\glsentryfmt}%
1442 }%
1443 {}%
```

Define sort counter if required:

```
1444 \@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.)

```
1445 \@ifnextchar[{\@gls@setcounter{#2}}%
1446   {\@gls@setcounter{#2}[\glscounter]}}%
1447 }
```

`\altnewglossary`

```
1448 \newcommand*{\altnewglossary}[3]{%
1449   \newglossary[#2-glg]{#1}{#2-gls}{#2-glo}{#3}%
1450 }
```

Only define new glossaries in the preamble:

```
1451 \@onlypreamble{\newglossary}
```

Only define new glossaries before `\makeglossaries`

```
1452 \@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`

```
1453 \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`

```
1454 \def\@gls@setcounter#1[#2]{%
1455   \expandafter\def\csname @gls@#1@counter\endcsname{#2}%
```

Add counter to xindy list, if not already added:

```
1456   \ifglxindy
1457     \GlsAddXdyCounters{#2}%
1458   \fi
1459 }
```

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

`\@gls@getcounter`

```
1460 \newcommand*{\@gls@getcounter}[1]{%
1461 \csname @gls@#1@counter\endcsname
1462 }
```

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

```
1463 \glsdefmain
```

Define the “acronym” glossaries if required.

```
1464 \@gls@do@acronymsdef
```

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

```
1465 \@gls@do@symbolsdef
1466 \@gls@do@numbersdef
1467 \@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.

```
1468 \define@key{glossentry}{name}{%
1469 \def\@glo@name{#1}%
1470 }
```

**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`.

```
1471 \define@key{glossentry}{description}{%
1472 \def\@glo@desc{#1}%
1473 }
```

**descriptionplural**

```
1474 \define@key{glossentry}{descriptionplural}{%
1475 \def\@glo@descplural{#1}%
1476 }
```

**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>`.

```
1477 \define@key{glossentry}{sort}{%
1478 \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.

```
1479 \define@key{glossentry}{text}{%
1480 \def\@glo@text{#1}%
1481 }
```

**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.

```
1482 \define@key{glossentry}{plural}{%
1483 \def\@glo@plural{#1}%
1484 }
```

**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.

```
1485 \define@key{glossentry}{first}{%
1486 \def\@glo@first{#1}%
1487 }
```

**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.

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

`\@gls@default@value`

```
1491 \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).

```
1492 \define@key{glossentry}{symbol}{%
1493 \def\@glo@symbol{#1}%
1494 }
```



symbolplural

```
1495 \define@key{glossentry}{symbolplural}{%
1496 \def\@glo@symbolplural{#1}%
1497 }
```

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

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

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

```
1500 \define@key{glossentry}{counter}{%
1501 \ifcsundef{c@#1}%
1502 {%
1503 \PackageError{glossaries}%
1504 {There is no counter called ‘#1’}%
1505 {%
1506 The counter key should have the name of a valid counter
1507 as its value%
1508 }%
1509 }%
1510 {%
1511 \def\@glo@counter{#1}%
1512 }%
1513 }
```

**see** The see key specifies a list of cross-references

```
1514 \define@key{glossentry}{see}{%
1515 \gls@checkseeallowed
1516 \def\@glo@see{#1}%
1517 \@glo@seeautonumberlist
1518 }
```

**gls@checkseeallowed**

```
1519 \newcommand*{\gls@checkseeallowed}{%
1520 \PackageError{glossaries}%
1521 {‘see’ key may only be used after \string\makeglossaries\space
1522 or \string\makenoidxglossaries}%
1523 {You must use \string\makeglossaries\space
1524 or \string\makenoidxglossaries\space before defining
1525 any entries that have a ‘see’ key}%
1526 }
```

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

```
1527 \define@key{glossentry}{parent}{%
1528 \def\@glo@parent{#1}}
```

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

```
1529 \define@choicekey{glossentry}{nonumberlist}[\val\nr]{true,false}[true]{%
1530   \ifcase\nr\relax
1531     \def\@glo@prefix{\glsnonextpages}%
1532   \else
1533     \def\@glo@prefix{\glsnextpages}%
1534   \fi
1535 }
```

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

user1

```
1536 \define@key{glossentry}{user1}{%
1537   \def\@glo@useri{#1}%
1538 }
```

user2

```
1539 \define@key{glossentry}{user2}{%
1540   \def\@glo@userii{#1}%
1541 }
```

user3

```
1542 \define@key{glossentry}{user3}{%
1543   \def\@glo@useriii{#1}%
1544 }
```

user4

```
1545 \define@key{glossentry}{user4}{%
1546   \def\@glo@useriv{#1}%
1547 }
```

user5

```
1548 \define@key{glossentry}{user5}{%
1549   \def\@glo@userv{#1}%
1550 }
```

user6

```
1551 \define@key{glossentry}{user6}{%
1552   \def\@glo@uservi{#1}%
1553 }
```

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.

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

shortplural This key is provided for use by \newacronym.

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

long This key is provided for use by \newacronym.

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

longplural This key is provided for use by \newacronym.

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

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

```
1566 \newcommand*\@glsnname{%
1567   \PackageError{glossaries}{name key required in
1568     \string\newglossaryentry\space for entry '\@glo@label'}{You
1569     haven't specified the entry name}}
```

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

```
1570 \newcommand*\@glsnodelsc{%
1571   \PackageError{glossaries}
1572   {%
1573     description key required in \string\newglossaryentry\space
1574     for entry '\@glo@label'%
1575   }%
1576   {%
1577     You haven't specified the entry description%
1578   }%
1579 }%
```

\@glsdefaultplural Now obsolete. Don't use.

```
1580 \newcommand*\@glsdefaultplural{}
```

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

```
1581 \newcommand*\@gls@missingnumberlist[1]{%
1582   ??%
1583   \ifglssavenumberlist
1584     \GlossariesWarning{Missing number list for entry '#1'.
1585       Maybe makeglossaries + rerun required.}%
1586   \else
1587     \PackageError{glossaries}%
1588     {Package option 'savenumberlist=true' required.}%
1589     {%
1590       You must use the 'savenumberlist' package option
1591       to reference location lists.%

```

```

1592 }%
1593 \fi
1594 }

```

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

```

1595 \newcommand*{\@glsdefaultsort}{\@glo@name}

```

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

```

1596 \newcount\gls@level

```

`@gls@noexpand@field`

```

1597 \newcommand{\@gls@noexpand@field}[3]{%
1598 \expandafter\global\expandafter
1599 \let\csname glo@#1@#2\endcsname#3%
1600 }

```

`gls@noexpand@fields`

```

1601 \newcommand{\@gls@noexpand@fields}[4]{%
1602 \ifcsdef{gls@assign@#3@field}
1603 {%
1604 \ifdefequal{#4}{\@gls@default@value}%
1605 {%
1606 \edef\@gls@value{\expandonce{#1}}%
1607 \csuse{gls@assign@#3@field}{#2}{\@gls@value}%
1608 }%
1609 {%
1610 \csuse{gls@assign@#3@field}{#2}{#4}%
1611 }%
1612 }%
1613 {%
1614 \ifdefequal{#4}{\@gls@default@value}%
1615 {%
1616 \edef\@gls@value{\expandonce{#1}}%
1617 \@gls@noexpand@field{#2}{#3}{\@gls@value}%
1618 }%
1619 {%
1620 \@gls@noexpand@field{#2}{#3}{#4}%
1621 }%
1622 }%
1623 }

```

`\@gls@expand@field`

```

1624 \newcommand{\@gls@expand@field}[3]{%
1625 \expandafter
1626 \protected@xdef\csname glo@#1@#2\endcsname{#3}%
1627 }

```

@gls@expand@fields

```

1628 \newcommand{\@gls@expand@fields}[4]{%
1629   \ifcsdef{gls@assign@#3@field}
1630   {%
1631     \ifdefequal{#4}{\@gls@default@value}%
1632     {%
1633       \edef\@gls@value{\expandonce{#1}}%
1634       \csuse{gls@assign@#3@field}{#2}{\@gls@value}%
1635     }%
1636     {%
1637       \expandafter\@gls@startswithexpandonce#4\relax\relax\gls@endcheck
1638       {%
1639         \@gls@expand@field{#2}{#3}{#4}%
1640       }%
1641       {%
1642         \csuse{gls@assign@#3@field}{#2}{#4}%
1643       }%
1644     }%
1645   }%
1646   {%
1647     \ifdefequal{#4}{\@gls@default@value}%
1648     {%
1649       \@gls@expand@field{#2}{#3}{#1}%
1650     }%
1651     {%
1652       \@gls@expand@field{#2}{#3}{#4}%
1653     }%
1654   }%
1655 }

```

startswithexpandonce

```

1656 \def\@gls@expandonce{\expandonce}
1657 \def\@gls@startswithexpandonce#1#2\gls@endcheck#3#4{%
1658   \def\@gls@tmp{#1}%
1659   \ifdefequal{\@gls@expandonce}{\@gls@tmp}{#3}{#4}%
1660 }

```

\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.

```
1661 \let\gls@assign@field\@gls@expand@fields
```

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

```
1662 \newcommand*\glsexpandfields{%
```

```

1663 \let\gls@assign@field\@gls@expand@fields
1664 }

\glsnoexpandfields Don't expand values when assigning fields (except for specific fields that are
                    overridden by \glssetexpandfield).
1665 \newcommand*\glsnoexpandfields{%
1666 \let\gls@assign@field\@gls@noexpand@fields
1667 }

\newglossaryentry Define \newglossaryentry {<label> } {<key-val list>}. There are two required
                  fields in <key-val list>: name (or parent) and description. (See above.)
1668 \newrobustcmd{\newglossaryentry}[2]{%
    Check to see if this glossary entry has already been defined:
1669 \glsdoifnoexists{#1}%
1670 {%
1671 \gls@defglossaryentry{#1}{#2}%
1672 }%
1673 }

\provideglossaryentry Like \newglossaryentry but does nothing if the entry has already been de-
                    fined.
1674 \newrobustcmd{\provideglossaryentry}[2]{%
1675 \ifglsentryexists{#1}%
1676 {}%
1677 {%
1678 \gls@defglossaryentry{#1}{#2}%
1679 }%
1680 }
1681 \@onlypreamble{\provideglossaryentry}

\new@glossaryentry For use in document environment.
1682 \newrobustcmd{\new@glossaryentry}[2]{%
1683 \ifundef\@gls@deffile
1684 {%
1685 \global\newwrite\@gls@deffile
1686 \immediate\openout\@gls@deffile=\jobname.glsdefs
1687 }%
1688 {}%
1689 \ifglsentryexists{#1}{}%
1690 {%
1691 \gls@defglossaryentry{#1}{#2}%
1692 }%
1693 \@gls@writedef{#1}%
1694 }
1695 \AtBeginDocument
1696 {
1697 \makeatletter
1698 \InputIfFileExists{\jobname.glsdefs}{\input}{\typeout{Glossary file \jobname.glsdefs not found.}}

```

```

1699 \makeatother
1700 \let\newglossaryentry\new@glossaryentry
1701 }
1702 \AtEndDocument{\ifdef\@gls@deffile{\closeout\@gls@deffile}{}}

```

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

```

1703 \newcommand*{\@gls@writedef}[1]{%
1704   \immediate\write\@gls@deffile
1705   {%
1706     \string\ifglsentryexists{#1}{}\expandafter\@gobble\string\%^~J%
1707     \expandafter\@gobble\string\{\expandafter\@gobble\string\%^~J%
1708     \string\gls@defglossaryentry{\glsdetoklabel{#1}}\expandafter
1709       \@gobble\string\%^~J%
1710     \expandafter\@gobble\string\{\expandafter\@gobble\string\%%
1711   }%

```

Write key value information:

```

1712   \@for\@gls@map:=\@gls@keymap\do
1713   {%
1714     \edef\glo@value{\expandafter\expandonce
1715       \csname glo@\glsdetoklabel{#1}@\expandafter
1716         \@secondoftwo\@gls@map\endcsname}%
1717     \@onelevel@sanitize\glo@value
1718     \immediate\write\@gls@deffile
1719     {%
1720       \expandafter\@firstoftwo\@gls@map
1721       =\expandafter\@gobble\string\{\glo@value\expandafter\@gobble\string\},%
1722       \expandafter\@gobble\string\%%
1723     }%
1724   }%

```

Provide hook:

```

1725   \gls@writedefhook
1726   \immediate\write\@gls@deffile
1727   {%
1728     \expandafter\@gobble\string\%^~J%
1729     \expandafter\@gobble\string\}\expandafter\@gobble\string\%^~J%
1730     \expandafter\@gobble\string\}\expandafter\@gobble\string\%%
1731   }%
1732 }

```

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

```

1733 \newcommand*{\@gls@keymap}{%
1734   {name}{name},%
1735   {sort}{sortvalue},% unescaped sort value
1736   {type}{type},%
1737   {first}{first},%
1738   {firstplural}{firstpl},%
1739   {text}{text},%

```

```

1740 {plural}{plural},%
1741 {description}{desc},%
1742 {descriptionplural}{descplural},%
1743 {symbol}{symbol},%
1744 {symbolplural}{symbolplural},%
1745 {user1}{useri},%
1746 {user2}{userii},%
1747 {user3}{useriii},%
1748 {user4}{useriv},%
1749 {user5}{userv},%
1750 {user6}{uservi},%
1751 {long}{long},%
1752 {longplural}{longpl},%
1753 {short}{short},%
1754 {shortplural}{shortpl},%
1755 {counter}{counter},%
1756 {parent}{parent}%
1757 }

```

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

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

```
1758 \newcommand*{\@gls@fetchfield}[2]{%
```

Ensure user field name is fully expanded

```
1759 \edef\@gls@thisval{#2}%
```

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

```

1760 \@for\@gls@map:=\@gls@keymap\do{%
1761 \edef\@this@key{\expandafter\@firstoftwo\@gls@map}%
1762 \ifdefequal{\@this@key}{\@gls@thisval}%
1763 {%

```

Found it.

```
1764 \edef#1{\expandafter\@secondoftwo\@gls@map}%
```

Break out of loop.

```

1765 \@endfortrue
1766 }%
1767 {}%
1768 }%
1769 }

```

`\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.

```
1770 \newcommand*{\glsaddkey}{\@ifstar\@sglsaddkey\@glsaddkey}
```



Starred version switches on expansion for this key.

```

1771 \newcommand*{\@sglsaddkey}[1]{%
1772   \key@ifundefined{glossentry}{#1}%
1773   {%
1774     \expandafter\newcommand\expandafter*\expandafter
1775     {\csname gls@assign@#1@field\endcsname}[2]{%
1776       \@gls@expand@field{##1}{#1}{##2}%
1777     }%
1778   }%
1779   }%
1780   \@glsaddkey{#1}%
1781 }
```

Unstarred version doesn't override default expansion.

```

1782 \newcommand*{\@glsaddkey}[7]{%
```

Check the specified key doesn't already exist.

```

1783   \key@ifundefined{glossentry}{#1}%
1784   {%
```

Set up the key.

```

1785     \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
1786     \appto\@gls@keymap{,{#1}{#1}}%
```

Set the default value.

```

1787     \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
```

Assignment code.

```

1788     \appto\@newglossaryentryposthook{%
1789       \letcs{\@glo@tmp}{@glo@#1}%
1790       \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
1791     }%
```

Define the no-link commands.

```

1792     \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
1793     \newcommand*{#4}[1]{\@Gls@entry@field{##1}{#1}}%
```

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

```

1794     \ifcsdef{@gls@user@#1}%
1795     {%
1796       \PackageError{glossaries}%
1797       {Can't define '\string#5' as helper command
1798       '\expandafter\string\csname @gls@user@#1\endcsname' already exists}%
1799       {}%
1800     }%
1801     {%
1802       \newrobustcmd*{#5}{\@ifstar{\csuse{@sgls@user@#1}}{\csuse{@gls@user@#1}}}%
1803       \expandafter\newcommand\expandafter*\expandafter
1804       {\csname @sgls@user@#1\endcsname}[1][1]{%
1805         \csuse{@gls@user@#1}[hyper=false,##1]%
1806       }%
1807       \expandafter\newcommand\expandafter*\expandafter
```

```

1808     {\csname @gls@user@#1\endcsname}[2] [] {%
1809         \new@ifnextchar [%
1810             {\csuse{@gls@user@#1@}{##1}{##2}}%
1811             {\csuse{@gls@user@#1@}{##1}{##2} []}}%
1812     \csdef{@gls@user@#1@}##1##2[##3] {%
1813         \@gls@field@link{##1}{##2}{#3{##2}##3}%
1814     }%
1815 }%

```

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

```

1816     \ifcsdef{@Gls@user@#1@}%
1817     {%
1818         \PackageError{glossaries}%
1819         {Can't define '\string#6' as helper command
1820         '\expandafter\string\csname @Gls@user@#1@\endcsname' already exists}%
1821     }%
1822 }%
1823 {%
1824     \newrobustcmd*{#6}{\@ifstar{\csuse{@sGls@user@#1}}{\csuse{@Gls@user@#1}}}%
1825     \expandafter\newcommand\expandafter*\expandafter
1826     {\csname @sGls@user@#1\endcsname}[1] [] {%
1827         \csuse{@Gls@user@#1}[hyper=false,##1]%
1828     }%
1829     \expandafter\newcommand\expandafter*\expandafter
1830     {\csname @Gls@user@#1\endcsname}[2] [] {%
1831         \new@ifnextchar [%
1832             {\csuse{@Gls@user@#1@}{##1}{##2}}%
1833             {\csuse{@Gls@user@#1@}{##1}{##2} []}}%
1834     \csdef{@Gls@user@#1@}##1##2[##3] {%
1835         \@gls@field@link{##1}{##2}{#4{##2}##3}%
1836     }%
1837 }%

```

Finally the all caps version:

```

1838     \ifcsdef{@GLS@user@#1@}%
1839     {%
1840         \PackageError{glossaries}%
1841         {Can't define '\string#7' as helper command
1842         '\expandafter\string\csname @GLS@user@#1@\endcsname' already exists}%
1843     }%
1844 }%
1845 {%
1846     \newrobustcmd*{#7}{\@ifstar{\csuse{@sGLS@user@#1}}{\csuse{@GLS@user@#1}}}%
1847     \expandafter\newcommand\expandafter*\expandafter
1848     {\csname @sGLS@user@#1\endcsname}[1] [] {%
1849         \csuse{@GLS@user@#1}[hyper=false,##1]%
1850     }%
1851     \expandafter\newcommand\expandafter*\expandafter
1852     {\csname @GLS@user@#1\endcsname}[2] [] {%
1853         \new@ifnextchar [%

```

```

1854         {\csuse{@GLS@user@#1@}{##1}{##2}}%
1855         {\csuse{@GLS@user@#1@}{##1}{##2}[]}}%
1856     \csdef{@GLS@user@#1@}##1##2[##3]{%
1857         \@gls@field@link{##1}{##2}{\mfirstucMakeUppercase{#3{##2}##3}}%
1858     }%
1859 }%
1860 }%
1861 {%
1862     \PackageError{glossaries}{Key ‘#1’ already exists}{}%
1863 }%
1864 }

```

`\gls.writedefhook`

```

1865 \newcommand*{\gls.writedefhook}{}

```

`\gls@assign@desc`

```

1866 \newcommand*{\gls@assign@desc}[1]{%
1867     \gls@assign@field{#1}{desc}{\@glo@desc}%
1868     \gls@assign@field{\@glo@desc}{#1}{descplural}{\@glo@descplural}%
1869 }

```

`\longnewglossaryentry`

```

1870 \newcommand{\longnewglossaryentry}[3]{%
1871     \glsdoifnoexists{#1}%
1872     {%
1873         \bgroup
1874         \let\@org@newglossaryentryprehook\@newglossaryentryprehook
1875         \long\def\@newglossaryentryprehook{%
1876             \long\def\@glo@desc{#3\leavevmode\unskip\nopostdesc}%
1877             \@org@newglossaryentryprehook
1878         }%
1879         \renewcommand*{\gls@assign@desc}[1]{%
1880             \global\cslet{glo@glstdetoklabel{#1}@desc}{\@glo@desc}%
1881             \global\cslet{glo@glstdetoklabel{#1}@descplural}{\@glo@desc}%
1882         }
1883         \gls@defglossaryentry{#1}{#2}%
1884         \egroup
1885     }
1886 }

```

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

```

1887 \@onlypreamble{\longnewglossaryentry}

```

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

```

1888 \newcommand{\longprovideglossaryentry}[3]{%
1889     \ifglstentryexists{#1}{}%
1890     {\longnewglossaryentry{#1}{#2}{#3}}%
1891 }
1892 \@onlypreamble{\longprovideglossaryentry}

```

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

Defines a new entry without checking if it already exists.

```
1893 \newcommand{\gls@defglossaryentry}[2]{%
```

Store label

```
1894 \edef\@glo@label{\glsdetoklabel{#1}}%
```

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

```
1895 \let\glslabel\@glo@label
```

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

```
1896 \let\@glo@name\@gls@name
```

```
1897 \let\@glo@desc\@gls@desc
```

```
1898 \let\@glo@descplural\@gls@default@value
```

```
1899 \let\@glo@type\@gls@default@value
```

```
1900 \let\@glo@symbol\@gls@default@value
```

```
1901 \let\@glo@symbolplural\@gls@default@value
```

```
1902 \let\@glo@text\@gls@default@value
```

```
1903 \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.)

```
1904 \let\@glo@first\@gls@default@value
```

```
1905 \let\@glo@firstplural\@gls@default@value
```

Set the default sort:

```
1906 \let\@glo@sort\@gls@default@value
```

Set the default counter:

```
1907 \let\@glo@counter\@gls@default@value
```

```
1908 \def\@glo@see{}%
```

```
1909 \def\@glo@parent{}%
```

```
1910 \def\@glo@prefix{}%
```

```
1911 \def\@glo@useri{}%
```

```
1912 \def\@glo@userii{}%
```

```
1913 \def\@glo@useriii{}%
```

```
1914 \def\@glo@useriv{}%
```

```
1915 \def\@glo@userv{}%
```

```
1916 \def\@glo@uservi{}%
```

```

1917 \def\@glo@short{}%
1918 \def\@glo@shortpl{}%
1919 \def\@glo@long{}%
1920 \def\@glo@longpl{}%

```

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

```

1921 \@newglossaryentryprehook

```

Extract key-val information from third parameter:

```

1922 \setkeys{glossentry}{#2}%

```

Check there is a default glossary.

```

1923 \ifundef\glsdefaulttype
1924 {%
1925   \PackageError{glossaries}%
1926   {No default glossary type (have you used ‘nomain’?)}%
1927   {If you use package option ‘nomain’ you must define
1928    a new glossary before you can define entries}%
1929 }%
1930 {}%

```

Assign type. This must be fully expandable

```

1931 \gls@assign@field{\glsdefaulttype}{\@glo@label}{type}{\@glo@type}%
1932 \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.

```

1933 \ifcsundef{glolist@\@glo@type}%
1934 {%
1935   \PackageError{glossaries}%
1936   {Glossary type ‘\@glo@type’ has not been defined}%
1937   {You need to define a new glossary type, before making entries
1938    in it}%
1939 }%
1940 {%
1941   \protected@edef\@glolist@{\csname glolist@\@glo@type\endcsname}%
1942   \expandafter\xdef\csname glolist@\@glo@type\endcsname{%
1943     \@glolist@{\@glo@label},}%
1944   }%

```

Initialise level to 0.

```

1945 \gls@level=0\relax

```

Has this entry been assigned a parent?

```

1946 \ifx\@glo@parent\@empty

```

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

```

1947 \expandafter\gdef\csname glo@\@glo@label @parent\endcsname{}%
1948 \else

```

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

```

1949 \ifdefequal\@glo@label\@glo@parent%
1950 {%

```

```

1951      \PackageError{glossaries}{Entry ‘\@glo@label’ can’t be its own parent}{}%
1952      \def\@glo@parent{}%
1953      \expandafter\gdef\csname glo@\@glo@label @parent\endcsname{%
1954      }%
1955      {%

```

Check the parent exists:

```

1956      \ifglentryexists{\@glo@parent}%
1957      {%

```

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

```

1958      \expandafter\xdef\csname glo@\@glo@label @parent\endcsname{%
1959      \@glo@parent}%

```

Determine level.

```

1960      \gls@level=\csname glo@\@glo@parent @level\endcsname\relax
1961      \advance\gls@level by 1\relax

```

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

```

1962      \ifx\@glo@name\@glsnoname
1963      \expandafter\let\expandafter\@glo@name
1964      \csname glo@\@glo@parent @name\endcsname

```

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

```

1965      \ifx\@glo@plural\@gls@default@value
1966      \expandafter\let\expandafter\@glo@plural
1967      \csname glo@\@glo@parent @plural\endcsname
1968      \fi
1969      \fi
1970      }%
1971      {%

```

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

```

1972      \PackageError{glossaries}%
1973      {%
1974      Invalid parent ‘\@glo@parent’
1975      for entry ‘\@glo@label’ - parent doesn’t exist%
1976      }%
1977      {%
1978      Parent entries must be defined before their children%
1979      }%
1980      \def\@glo@parent{}%
1981      \expandafter\gdef\csname glo@\@glo@label @parent\endcsname{%
1982      }%
1983      }%
1984      \fi

```

Set the level for this entry

```

1985      \expandafter\xdef\csname glo@\@glo@label @level\endcsname{\number\gls@level}%

```

Define commands associated with this entry:

```

1986 \gls@assign@field{\@glo@name}{\@glo@label}{sortvalue}{\@glo@sort}%
1987 \letcs\@glo@sort{glo@\@glo@label @sortvalue}%
1988 \gls@assign@field{\@glo@name}{\@glo@label}{text}{\@glo@text}%
1989 \expandafter\gls@assign@field\expandafter
1990 {\csname glo@\@glo@label @text\endcsname\glspluralsuffix}%
1991 {\@glo@label}{plural}{\@glo@plural}%
1992 \expandafter\gls@assign@field\expandafter
1993 {\csname glo@\@glo@label @text\endcsname}%
1994 {\@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.

1995 \ifx\@glo@first\@gls@default@value
1996 \expandafter\gls@assign@field\expandafter
1997 {\csname glo@\@glo@label @plural\endcsname}%
1998 {\@glo@label}{firstpl}{\@glo@firstplural}%
1999 \else
2000 \expandafter\gls@assign@field\expandafter
2001 {\csname glo@\@glo@label @first\endcsname\glspluralsuffix}%
2002 {\@glo@label}{firstpl}{\@glo@firstplural}%
2003 \fi

2004 \ifcsundef{@glo@type@\@glo@type @counter}%
2005 {%
2006 \def\@glo@defaultcounter{\glscounter}%
2007 }%
2008 {%
2009 \letcs\@glo@defaultcounter{@glo@type@\@glo@type @counter}%
2010 }%
2011 \gls@assign@field{\@glo@defaultcounter}{\@glo@label}{counter}{\@glo@counter}%
2012 \gls@assign@field{}{\@glo@label}{useri}{\@glo@useri}%
2013 \gls@assign@field{}{\@glo@label}{userii}{\@glo@userii}%
2014 \gls@assign@field{}{\@glo@label}{useriii}{\@glo@useriii}%
2015 \gls@assign@field{}{\@glo@label}{useriv}{\@glo@useriv}%
2016 \gls@assign@field{}{\@glo@label}{userv}{\@glo@userv}%
2017 \gls@assign@field{}{\@glo@label}{uservi}{\@glo@uservi}%
2018 \gls@assign@field{}{\@glo@label}{short}{\@glo@short}%
2019 \gls@assign@field{}{\@glo@label}{shortpl}{\@glo@shortpl}%
2020 \gls@assign@field{}{\@glo@label}{long}{\@glo@long}%
2021 \gls@assign@field{}{\@glo@label}{longpl}{\@glo@longpl}%
2022 \ifx\@glo@name\@gls@noname
2023 \@gls@noname
2024 \let\@glo@name\@gls@default@value
2025 \fi
2026 \gls@assign@field{}{\@glo@label}{name}{\@glo@name}%

    Set default numberlist if not defined:

2027 \ifcsundef{glo@\@glo@label @numberlist}%
2028 {%
2029 \csxdef{glo@\@glo@label @numberlist}{%

```

```

2030      \noexpand\@gls@missingnumberlist{\@glo@label}}%
2031  }%
2032  {%

```

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.

```

2033  \def\@glo@@desc{\@glo@first}%
2034  \ifx\@glo@desc\@glo@@desc
2035    \let\@glo@desc\@glo@first
2036  \fi
2037  \ifx\@glo@desc\@glsnodesc
2038    \@glsnodesc
2039    \let\@glo@desc\@gls@default@value
2040  \fi
2041  \gls@assign@desc{\@glo@label}%

```

Set the sort key for this entry:

```

2042  \@gls@defsort{\@glo@type}{\@glo@label}%

2043  \def\@glo@@symbol{\@glo@text}%
2044  \ifx\@glo@symbol\@glo@@symbol
2045    \let\@glo@symbol\@glo@text
2046  \fi
2047  \gls@assign@field{\relax}{\@glo@label}{symbol}{\@glo@symbol}%
2048  \expandafter
2049    \gls@assign@field\expandafter
2050    {\csname glo@\@glo@label @symbol\endcsname}
2051    {\@glo@label}{symbolplural}{\@glo@symbolplural}%

```

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

```

2052  \expandafter\xdef\csname glo@\@glo@label @flagfalse\endcsname{%
2053    \noexpand\global
2054    \noexpand\let\expandafter\noexpand
2055    \csname ifglo@\@glo@label @flag\endcsname\noexpand\iffalse
2056  }%
2057  \expandafter\xdef\csname glo@\@glo@label @flagtrue\endcsname{%
2058    \noexpand\global
2059    \noexpand\let\expandafter\noexpand
2060    \csname ifglo@\@glo@label @flag\endcsname\noexpand\iftrue
2061  }%
2062  \csname glo@\@glo@label @flagfalse\endcsname

```

Sort out any cross-referencing if required.

```

2063  \ifdefined\@glo@see
2064  {%
2065  {%
2066    \protected@edef\@do@glsee{%
2067      \noexpand\@gls@fixbraces\noexpand\@glo@list\@glo@see
2068      \noexpand\@nil

```



```

2069      \noexpand\expandafter\noexpand\@glssee\noexpand\@glo@list{\@glo@label}}}%
2070      \@do@glssee
2071  }%

Determine and store main part of the entry's index format.
2072  \do@glo@storeentry{\@glo@label}%

Add end hook in case another package wants to add extra keys.
2073  \@newglossaryentryposthook
2074 }

\glossaryentryprehook Allow extra information to be added to glossary entries:
2075 \newcommand*{\@newglossaryentryprehook}{}

\glossaryentryposthook Allow extra information to be added to glossary entries:
2076 \newcommand*{\@newglossaryentryposthook}{}

\glsmoveentry Moves entry whose label is given by first argument to the glossary named in the
second argument.
2077 \newcommand*{\glsmoveentry}[2]{%
2078   \edef\@glo@thislabel{\glsdetoklabel{#1}}}%
2079   \edef\@glo@type{\csname glo@\@glo@thislabel @type\endcsname}%
2080   \def\@glo@list{,%}%
2081   \for@gl@entries[\@glo@type]{\@glo@label}%
2082   {%

2083     \ifdefequal\@glo@thislabel\@glo@label
2084       {\eappto\@glo@list{\@glo@label,}}}%
2085     }%
2086     \cslet{glolist@\@glo@type}{\@glo@list}%
2087     \csdef{glo@\@glo@thislabel @type}{#2}%
2088 }

@glossaryentryfield Indicate what command should be used to display each entry in the glossary.
(This enables the glossaries-accsupp package to use \accsuppglossaryentryfield
instead.)
2089 \ifglxindy
2090   \newcommand*{\@glossaryentryfield}{\string\glossentry}
2091 \else
2092   \newcommand*{\@glossaryentryfield}{\string\glossentry}
2093 \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.)
2094 \ifglxindy
2095   \newcommand*{\@glossarysubentryfield}{%
2096     \string\subglossentry}
2097 \else

```

```

2098 \newcommand*{\@glossarysubentryfield}{%
2099 \string\subglossentry}
2100 \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.)

```
2101 \newcommand{\@glo@storeentry}[1]{%
```

Escape makeindex/xindy special characters in the label:

```

2102 \edef\@glo@esclabel{#1}%
2103 \@gls@checkmkidxchars\@glo@esclabel

```

Get the sort string and escape any special characters

```

2104 \protected@edef\@glo@sort{\csname glo@#1@sort\endcsname}%
2105 \@gls@checkmkidxchars\@glo@sort

```

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

```
2106 \@gls@checkmkidxchars\@glo@prefix
```

Get the parent, if one exists

```
2107 \edef\@glo@parent{\csname glo@#1@parent\endcsname}%
```

Write the information to the glossary file.

```
2108 \ifglsxindy
```

Store using xindy syntax.

```
2109 \ifx\@glo@parent\@empty
```

Entry doesn't have a parent

```

2110 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
2111 (\string"\@glo@sort\string" %
2112 \string"\@glo@prefix\@glossarysubentryfield{\@glo@esclabel}\string") %
2113 }%
2114 \else

```

Entry has a parent

```

2115 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
2116 \csname glo@\@glo@parent @index\endcsname
2117 (\string"\@glo@sort\string" %
2118 \string"\@glo@prefix\@glossarysubentryfield
2119 {\csname glo@#1@level\endcsname}{\@glo@esclabel}\string") %
2120 }%
2121 \fi
2122 \else

```

Store using makeindex syntax.

```
2123 \ifx\@glo@parent\@empty
```

```

Sanitize \@glo@prefix
2124   \@onelevel@sanitize\@glo@prefix
Entry doesn't have a parent
2125   \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
2126     \@glo@sort\@gls@actualchar\@glo@prefix
2127     \@glossaryentryfield{\@glo@esclabel}%
2128   }%
2129   \else
Entry has a parent
2130   \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
2131     \csname glo@\@glo@parent @index\endcsname\@gls@levelchar
2132     \@glo@sort\@gls@actualchar\@glo@prefix
2133     \@glossarysubentryfield
2134       {\csname glo@#1@level\endcsname}{\@glo@esclabel}%
2135   }%
2136   \fi
2137 \fi
2138 }

```

## 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
2139 \AtBeginDocument{%
2140   \@ifpackageloaded{amsmath}%
2141   {\let\gls@ifnotmeasuring\@gls@ifnotmeasuring}%
2142   }%
2143 }
2144 \newcommand*\@gls@ifnotmeasuring[1]{%
2145   \ifmeasuring@
2146   \else
2147     #1%
2148   \fi
2149 }
2150 \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.

```

2151 \newcommand*\glsreset[1]{%
2152   \gls@ifnotmeasuring
2153   {%
2154     \glsdoifexists{#1}%

```

```

2155     {%
2156         \expandafter\global\csname glo@\glsdetoklabel{#1}@flagfalse\endcsname
2157     }%
2158 }%
2159 }

```

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

```

2160 \newcommand*{\glslocalreset}[1]{%
2161     \gls@ifnotmeasuring
2162     {%
2163         \glsdoifexists{#1}%
2164         {%
2165             \expandafter\let\csname ifglo@\glsdetoklabel{#1}@flag\endcsname\iffalse
2166         }%
2167     }%
2168 }

```

`\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.

```

2169 \newcommand*{\glsunset}[1]{%
2170     \gls@ifnotmeasuring
2171     {%
2172         \glsdoifexists{#1}%
2173         {%
2174             \expandafter\global\csname glo@\glsdetoklabel{#1}@flagtrue\endcsname
2175         }%
2176     }%
2177 }

```

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

```

2178 \newcommand*{\glslocalunset}[1]{%
2179     \gls@ifnotmeasuring
2180     {%
2181         \glsdoifexists{#1}%
2182         {%
2183             \expandafter\let\csname ifglo@\glsdetoklabel{#1}@flag\endcsname\iftrue
2184         }%
2185     }%
2186 }

```

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

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

`\glsresetall`

```

2187 \newcommand*{\glsresetall}[1][\@glo@types]{%
2188     \forallglsentries[#1]{\@glsentry}%
2189     {%
2190         \glsreset{\@glsentry}%
2191     }%
2192 }

```

As above, but with only a local effect:

`\glslocalresetall`

```
2193 \newcommand*{\glslocalresetall}[1][\@glo@types]{%
2194   \forallglsentries[#1]{\@glsentry}%
2195   {%
2196     \glslocalreset{\@glsentry}%
2197   }%
2198 }
```

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

Syntax: `\glsunsetall` [*<glossary-list>*]

`\glsunsetall`

```
2199 \newcommand*{\glsunsetall}[1][\@glo@types]{%
2200   \forallglsentries[#1]{\@glsentry}%
2201   {%
2202     \glsunset{\@glsentry}%
2203   }%
2204 }
```

As above, but with only a local effect:

`\glslocalunsetall`

```
2205 \newcommand*{\glslocalunsetall}[1][\@glo@types]{%
2206   \forallglsentries[#1]{\@glsentry}%
2207   {%
2208     \glslocalunset{\@glsentry}%
2209   }%
2210 }
```

## 1.9 Loading files containing glossary entries

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

`\loadglsentries` [*<type>*] [*<filename>*]

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

`\loadglsentries`

```
2211 \newcommand*{\loadglsentries}[2][\@gls@default]{%
```

---

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

```

2212 \let\@gls@default\glsdefaulttype
2213 \def\glsdefaulttype{#1}\input{#2}%
2214 \let\glsdefaulttype\@gls@default
2215 }

```

\loadglsentries can only be used in the preamble:

```

2216 \@onlypreamble{\loadglsentries}

```

## 1.10 Using glossary entries in the text

Any term that has been defined using \newglossaryentry (or \newacronym) can be displayed in the text (i.e. outside of the glossary) using one of the commands defined in this section. Unless you use \glslink, the way the term appears in the text is determined by \glsdisplayfirst (if it is the first time the term has been used) or \glsdisplay (for subsequent use). Any formatting commands (such as \textbf is governed by \glstextformat. By default this just displays the link text “as is”.

\glstextformat

```

2217 \newcommand*{\glstextformat}[1]{#1}

```

\glsentryfmt As from version 3.11a, the way in which an entry is displayed is now governed by \glsentryfmt. This doesn't take any arguments. The required information is set by commands like \gls. To ensure backward compatibility, the default use the old \glsdisplay and \glsdisplayfirst style of commands

```

2218 \newcommand*{\glsentryfmt}{%
2219 \@@gls@default@entryfmt\glsdisplayfirst\glsdisplay
2220 }

```

Format that provides backwards compatibility:

```

2221 \newcommand*{\@@gls@default@entryfmt}[2]{%
2222 \ifdefempty\glscustomtext
2223 {%
2224 \glsifplural
2225 {%

```

Plural form

```

2226 \glscapscase
2227 {%

```

Don't adjust case

```

2228 \ifglsused\glslabel
2229 {%

```

Subsequent use

```

2230 #2{\glsentryplural{\glslabel}}%
2231 {\glsentrydescplural{\glslabel}}%
2232 {\glsentrysymbolplural{\glslabel}}{\glsinsert}%
2233 }%
2234 {%

```

First use

```

2235      #1{\glentryfirstplural{\glslabel}}%
2236      {\glentrydescplural{\glslabel}}%
2237      {\glentrysymbolplural{\glslabel}}{\glinsert}%
2238    }%
2239  }%
2240  {%

```

Make first letter upper case

```

2241      \ifglused\glslabel
2242      {%

```

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

```

2243      \ifbool{glcompatible-3.07}%
2244      {%
2245        \protected@edef\@glo@etext{%
2246          #2{\glentryplural{\glslabel}}%
2247          {\glentrydescplural{\glslabel}}%
2248          {\glentrysymbolplural{\glslabel}}{\glinsert}}%
2249        \xmakefirstuc\@glo@etext
2250      }%
2251      {%
2252        #2{\Glentryplural{\glslabel}}%
2253        {\glentrydescplural{\glslabel}}%
2254        {\glentrysymbolplural{\glslabel}}{\glinsert}%
2255      }%
2256    }%
2257    {%

```

First use

```

2258      \ifbool{glcompatible-3.07}%
2259      {%
2260        \protected@edef\@glo@etext{%
2261          #1{\glentryfirstplural{\glslabel}}%
2262          {\glentrydescplural{\glslabel}}%
2263          {\glentrysymbolplural{\glslabel}}{\glinsert}}%
2264        \xmakefirstuc\@glo@etext
2265      }%
2266      {%
2267        #1{\Glentryfirstplural{\glslabel}}%
2268        {\glentrydescplural{\glslabel}}%
2269        {\glentrysymbolplural{\glslabel}}{\glinsert}%
2270      }%
2271    }%
2272  }%
2273  {%

```

### Make all upper case

```
2274      \ifglsused\glslabel
2275      {%
```

### Subsequent use

```
2276      \mfirstucMakeUppercase{#2{\glsentryplural{\glslabel}}}%
2277      {\glsentrydescplural{\glslabel}}}%
2278      {\glsentrysymbolplural{\glslabel}}{\glsinsert}}}%
2279      }%
2280      {%
```

### First use

```
2281      \mfirstucMakeUppercase{#1{\glsentryfirstplural{\glslabel}}}%
2282      {\glsentrydescplural{\glslabel}}}%
2283      {\glsentrysymbolplural{\glslabel}}{\glsinsert}}}%
2284      }%
2285      }%
2286      }%
2287      {%
```

### Singular form

```
2288      \glscapscase
2289      {%
```

### Don't adjust case

```
2290      \ifglsused\glslabel
2291      {%
```

### Subsequent use

```
2292      #2{\glsentrytext{\glslabel}}}%
2293      {\glsentrydesc{\glslabel}}}%
2294      {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2295      }%
2296      {%
```

### First use

```
2297      #1{\glsentryfirst{\glslabel}}}%
2298      {\glsentrydesc{\glslabel}}}%
2299      {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2300      }%
2301      }%
2302      {%
```

### Make first letter upper case

```
2303      \ifglsused\glslabel
2304      {%
```

### Subsequent use

```
2305      \ifbool{glscompatible-3.07}%
2306      {%
2307      \protected@edef\@glo@etext{%
2308      #2{\glsentrytext{\glslabel}}}%
```



```

2309         {\glsentrydesc{\glslabel}}}%
2310         {\glsentrysymbol{\glslabel}}{\glsinsert}}}%
2311     \xmakefirstuc\@glo@etext
2312 }%
2313 {%
2314     #2{\Glsentrytext{\glslabel}}}%
2315     {\glsentrydesc{\glslabel}}}%
2316     {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2317 }%
2318 }%
2319 {%

```

#### First use

```

2320     \ifbool{glscompatible-3.07}%
2321     {%
2322         \protected@edef\@glo@etext{%
2323             #1{\glsentryfirst{\glslabel}}}%
2324             {\glsentrydesc{\glslabel}}}%
2325             {\glsentrysymbol{\glslabel}}{\glsinsert}}}%
2326         \xmakefirstuc\@glo@etext
2327     }%
2328     {%
2329         #1{\Glsentryfirst{\glslabel}}}%
2330         {\glsentrydesc{\glslabel}}}%
2331         {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2332     }%
2333 }%
2334 }%
2335 {%

```

#### Make all upper case

```

2336     \ifglsused\glslabel
2337     {%

```

#### Subsequent use

```

2338         \mfirstucMakeUppercase{#2{\glsentrytext{\glslabel}}}%
2339         {\glsentrydesc{\glslabel}}}%
2340         {\glsentrysymbol{\glslabel}}{\glsinsert}}}%
2341     }%
2342     {%

```

#### First use

```

2343         \mfirstucMakeUppercase{#1{\glsentryfirst{\glslabel}}}%
2344         {\glsentrydesc{\glslabel}}}%
2345         {\glsentrysymbol{\glslabel}}{\glsinsert}}}%
2346     }%
2347 }%
2348 }%
2349 }%
2350 {%

```

Custom text provided in \glsdisp

```
2351 \ifglsused{\glslabel}%  
2352 {%
```

Subsequent use

```
2353 #2{\glscustomtext}%  
2354 {\glsentrydesc{\glslabel}}%  
2355 {\glsentrysymbol{\glslabel}}{}%  
2356 }%  
2357 {%
```

First use

```
2358 #1{\glscustomtext}%  
2359 {\glsentrydesc{\glslabel}}%  
2360 {\glsentrysymbol{\glslabel}}{}%  
2361 }%  
2362 }%  
2363 }
```

`\glsentryfmt` Define a generic format that just uses the first, text, plural or first plural keys (or the custom text) with the insert text appended.

```
2364 \newcommand*{\glsentryfmt}{%  
2365 \ifdefempty\glscustomtext  
2366 {%  
2367 \glsifplural  
2368 {%
```

Plural form

```
2369 \glscapscase  
2370 {%
```

Don't adjust case

```
2371 \ifglsused\glslabel  
2372 {%
```

Subsequent use

```
2373 \glsentryplural{\glslabel}\glsinsert  
2374 }%  
2375 {%
```

First use

```
2376 \glsentryfirstplural{\glslabel}\glsinsert  
2377 }%  
2378 }%  
2379 {%
```

Make first letter upper case

```
2380 \ifglsused\glslabel  
2381 {%
```

Subsequent use.

```
2382 \Glsentryplural{\glslabel}\glsinsert
```

2383       }%  
 2384       {%

First use

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

Make all upper case

2389       \ifglsused\glslabel  
 2390       {%

Subsequent use

2391       \mfirstucMakeUppercase  
 2392       {\glsentryplural{\glslabel}\glsinsert}%  
 2393       }%  
 2394       {%

First use

2395       \mfirstucMakeUppercase  
 2396       {\glsentryfirstplural{\glslabel}\glsinsert}%  
 2397       }%  
 2398       }%  
 2399       }%  
 2400       {%

Singular form

2401       \glscapscase  
 2402       {%

Don't adjust case

2403       \ifglsused\glslabel  
 2404       {%

Subsequent use

2405       \glsentrytext{\glslabel}\glsinsert  
 2406       }%  
 2407       {%

First use

2408       \glsentryfirst{\glslabel}\glsinsert  
 2409       }%  
 2410       }%  
 2411       {%

Make first letter upper case

2412       \ifglsused\glslabel  
 2413       {%

Subsequent use

2414       \Glsentrytext{\glslabel}\glsinsert  
 2415       }%  
 2416       {%

First use

```
2417      \Glsentryfirst{\glslabel}\glsinsert
2418      }%
2419      }%
2420      {%
```

Make all upper case

```
2421      \ifglsused\glslabel
2422      {%
```

Subsequent use

```
2423      \mfirstucMakeUppercase{\glsentrytext{\glslabel}\glsinsert}%
2424      }%
2425      {%
```

First use

```
2426      \mfirstucMakeUppercase{\glsentryfirst{\glslabel}\glsinsert}%
2427      }%
2428      }%
2429      }%
2430      }%
2431      {%
```

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

```
2432      \glscustomtext\glsinsert
2433      }%
2434 }
```

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

```
2435 \newcommand*{\glsngenacfmt}{%
2436   \ifdefempty\glscustomtext
2437   {%
2438     \ifglsused\glslabel
2439     {%
```

Subsequent use:

```
2440     \glsifplural
2441     {%
```

Subsequent plural form:

```
2442     \glscapscase
2443     {%
```

Subsequent plural form, don't adjust case:

```
2444     \acronymfont{\glsentryshortpl{\glslabel}}\glsinsert
2445     }%
2446     {%
```

Subsequent plural form, make first letter upper case:

```
2447      \acronymfont{\Glsentryshortpl{\glslabel}}\glsinsert
2448      }%
2449      {%
```

Subsequent plural form, all caps:

```
2450      \mfirstucMakeUppercase
2451      {\acronymfont{\Glsentryshortpl{\glslabel}}\glsinsert}%
2452      }%
2453      }%
2454      {%
```

Subsequent singular form

```
2455      \glscapscase
2456      {%
```

Subsequent singular form, don't adjust case:

```
2457      \acronymfont{\Glsentryshort{\glslabel}}\glsinsert
2458      }%
2459      {%
```

Subsequent singular form, make first letter upper case:

```
2460      \acronymfont{\Glsentryshort{\glslabel}}\glsinsert
2461      }%
2462      {%
```

Subsequent singular form, all caps:

```
2463      \mfirstucMakeUppercase
2464      {\acronymfont{\Glsentryshort{\glslabel}}\glsinsert}%
2465      }%
2466      }%
2467      }%
2468      {%
```

First use:

```
2469      \glsifplural
2470      {%
```

First use plural form:

```
2471      \glscapscase
2472      {%
```

First use plural form, don't adjust case:

```
2473      \genplacrfullformat{\glslabel}{\glsinsert}%
2474      }%
2475      {%
```

First use plural form, make first letter upper case:

```
2476      \Genplacrfullformat{\glslabel}{\glsinsert}%
2477      }%
2478      {%
```

First use plural form, all caps:

```
2479      \mfirstucMakeUppercase
2480      {\genplacrfullformat{\glslabel}{\glsinsert}}%
2481      }%
2482      }%
2483      {%
```

First use singular form

```
2484      \glscapscase
2485      {%
```

First use singular form, don't adjust case:

```
2486      \genacrfullformat{\glslabel}{\glsinsert}%
2487      }%
2488      {%
```

First use singular form, make first letter upper case:

```
2489      \Genacrfullformat{\glslabel}{\glsinsert}%
2490      }%
2491      {%
```

First use singular form, all caps:

```
2492      \mfirstucMakeUppercase
2493      {\genacrfullformat{\glslabel}{\glsinsert}}%
2494      }%
2495      }%
2496      }%
2497      }%
2498      {%
```

User supplied text.

```
2499      \glscustomtext
2500      }%
2501 }
```

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

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

```
2502 \newcommand*{\genacrfullformat}[2]{%
2503   \glsentrylong{#1}#2\space
2504   (\protect\firstacronymfont{\glsentryshort{#1}})%
2505 }
```

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

As above but makes the first letter upper case.

```
2506 \newcommand*{\Genacrfullformat}[2]{%
2507   \protected@edef\gls@text{\genacrfullformat{#1}{#2}}%
```

```

2508 \xmakefirstuc\gls@text
2509 }

```

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

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

```

2510 \newcommand*{\genplacrfullformat}[2]{%
2511   \glsentrylongpl{#1}#2\space
2512   (\protect\firstacronymfont{\glsentryshortpl{#1}})%
2513 }

```

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

As above but makes the first letter upper case.

```

2514 \newcommand*{\Genplacrfullformat}[2]{%
2515   \protected@edef\gls@text{\genplacrfullformat{#1}{#2}}%
2516   \xmakefirstuc\gls@text
2517 }

```

`\glsdisplayfirst` Deprecated. Kept for backward compatibility.

```

2518 \newcommand*{\glsdisplayfirst}[4]{#1#4}

```

`\glsdisplay` Deprecated. Kept for backward compatibility.

```

2519 \newcommand*{\glsdisplay}[4]{#1#4}

```

`\defglsdisplay` Deprecated. Kept for backward compatibility.

```

2520 \newcommand*{\defglsdisplay}[2][\glsdefaulttype]{%
2521   \GlossariesWarning{\string\defglsdisplay\space is now obsolete.^^J
2522   Use \string\defglsentryfmt\space instead}%
2523   \expandafter\def\csname gls@#1@display\endcsname##1##2##3##4{#2}%
2524   \edef\@gls@doentrydef{%
2525     \noexpand\defglsentryfmt[#1]{%
2526       \noexpand\ifcsdef{gls@#1@displayfirst}%
2527       {%
2528         \noexpand\@gls@default@entryfmt
2529         {\noexpand\csuse{gls@#1@displayfirst}}%
2530         {\noexpand\csuse{gls@#1@display}}%
2531       }%
2532       {%
2533         \noexpand\@gls@default@entryfmt
2534         {\noexpand\glsdisplayfirst}%
2535         {\noexpand\csuse{gls@#1@display}}%
2536       }%
2537     }%
2538   }%
2539   \@gls@doentrydef
2540 }

```

`\defglsdisplayfirst` Deprecated. Kept for backward compatibility.

```

2541 \newcommand*{\defglsdisplayfirst}[2][\glsdefaultttype]{%
2542   \GlossariesWarning{\string\defglsdisplayfirst\space is now obsolete.^^J
2543   Use \string\defglsentryfmt\space instead}%
2544   \expandafter\def\csname gls@#1@displayfirst\endcsname##1##2##3##4{#2}%
2545   \edef\@gls@doentrydef{%
2546     \noexpand\defglsentryfmt[#1]{%
2547       \noexpand\ifcsdef{gls@#1@display}%
2548       {%
2549         \noexpand\@gls@default@entryfmt
2550         {\noexpand\csuse{gls@#1@displayfirst}}}%
2551         {\noexpand\csuse{gls@#1@display}}}%
2552       }%
2553       {%
2554         \noexpand\@gls@default@entryfmt
2555         {\noexpand\csuse{gls@#1@displayfirst}}}%
2556         {\noexpand\glsdisplay}}%
2557     }%
2558   }%
2559 }%
2560 \@gls@doentrydef
2561 }

```

### 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.

```

2562 \define@key{glslink}{counter}{%
2563   \ifcsundef{c@#1}%
2564   {%
2565     \PackageError{glossaries}%
2566     {There is no counter called ‘#1’}%
2567     {%
2568       The counter key should have the name of a valid counter
2569       as its value%

```



```

2570 }%
2571 }%
2572 {%
2573 \def\@gls@counter{#1}%
2574 }%
2575 }

```

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.

```

2576 \define@key{glslink}{format}{%
2577 \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.

```

2578 \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.

```

2579 \define@boolkey{glslink}{local}[true]{}

```

Syntax:

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

Display *<text>* in the document, and add the entry information for *<label>* into the relevant glossary. The optional argument should be a key value list using the glslink keys defined above.

There is also a starred version:

```
\glslink*[<options>]{<label>}{<text>}
```

which is equivalent to \glslink[hyper=false,*<options>*]{*<label>*}{*<text>*}

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

```
\glslink
```

```

2580 \newrobustcmd*{\glslink}{%
2581 \@ifstar\@sgls@link\@gls@@link
2582 }

```

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

```

2583 \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.

```

2584 \newcommand*{\@gls@link}[3][]{%
2585   \ifglsentryexists{#2}%
2586   {%
2587     \@gls@link[#1]{#2}{#3}%
2588   }{%
2589     \PackageError{glossaries}{Glossary entry ‘#2’ has not been
2590     defined}{You need to define a glossary entry before you
2591     can use it.}%

```

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

```

2592   \glstextformat{#3}%
2593 }%
2594 }

```

`\@gls@link`

```

2595 \def\@gls@link[#1]#2#3{%

```

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

```

2596   \leavevmode
2597   \edef\glslabel{\glsdetoklabel{#2}}%

```

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

```

2598   \def\@gls@link@opts{#1}%
2599   \let\@gls@link@label\glslabel
2600   \def\@glsnumberformat{\glsnumberformat}%
2601   \edef\@gls@counter{\csname glo@\glslabel @counter\endcsname}%

```

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

```

2602   \edef\gls@type{\csname glo@\glslabel @type\endcsname}%
2603   \expandafter\DTLifinlist\expandafter
2604     {\gls@type}{\@gls@nohyperlist}%
2605   {%
2606     \KV@glslink@hyperfalse
2607   }%
2608   {%
2609     \KV@glslink@hypertrue
2610   }%
2611   \setkeys{glslink}{#1}%

```

Store the entry's counter in `\theglsentrycounter`

```

2612   \@gls@saveentrycounter

```

Define sort key if necessary:

```

2613   \@gls@setsort{\glslabel}%

```

(De-tok'ing done by `\@do@wrglossary`)

```

2614   \@do@wrglossary{#2}%
2615   \ifKV@glslink@hyper

```

```

2616      \@glslink{\glolinkprefix\glslabel}{\glstextformat{#3}}%
2617      \else

2618      \glstextformat{#3}%
2619      \fi
2620 }

\glolinkprefix
2621 \newcommand*{\glolinkprefix}{glo:}

\glsentrycounter    Set default value of entry counter
2622 \def\glsentrycounter{\glscounter}%

\ls@saveentrycounter    Need to check if using equation counter in align environment:
2623 \newcommand*{\@gls@saveentrycounter}{%
2624   \def\@gls@Hcounter{%
      Are we using equation counter?
2625     \ifthenelse{\equal{\@gls@counter}{equation}}{%
2626       {
          If we're in align environment, \xatlevel@ will be defined. (Can't test for
          \@currentvir as may be inside an inner environment.)
2627       \ifcsundef{xatlevel@}%
2628       {%
2629         \edef\theglsentrycounter{\expandafter\noexpand
2630           \csname the\@gls@counter\endcsname}%
2631       }%
2632       {%
2633         \ifx\xatlevel@\@empty
2634           \edef\theglsentrycounter{\expandafter\noexpand
2635             \csname the\@gls@counter\endcsname}%
2636         \else
2637           \savecounters@
2638           \advance\c@equation by 1\relax
2639           \edef\theglsentrycounter{\csname the\@gls@counter\endcsname}%
          Check if hyperref version of this counter
2640           \ifcsundef{theH\@gls@counter}%
2641           {%
2642             \def\@gls@Hcounter{\theglsentrycounter}%
2643           }%
2644           {%
2645             \def\@gls@Hcounter{\csname theH\@gls@counter\endcsname}%
2646           }%
2647           \protected@edef\theHglentrycounter{\@gls@Hcounter}%
2648           \restorecounters@
2649         \fi
2650       }%
2651     }%
2652   }%

```

Not using equation counter so no special measures:

```
2653 \edef\theglsentrycounter{\expandafter\noexpand
2654 \csname the\@gls@counter\endcsname}%
2655 }%
```

Check if hyperref version of this counter

```
2656 \ifx\@gls@Hcounter\@empty
2657 \ifcsundef{theH\@gls@counter}%
2658 {%
2659 \def\theHglentrycounter{\theglsentrycounter}%
2660 }%
2661 {%
2662 \protected@edef\theHglentrycounter{\expandafter\noexpand
2663 \csname theH\@gls@counter\endcsname}%
2664 }%
2665 \fi
2666 }
```

`\@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.

```
2667 \def\@set@glo@numformat#1#2#3#4{%
2668 \expandafter\@glo@check@mkidxrangechar#3\@nil
2669 \protected@edef#1{%
2670 \@glo@prefix setentrycounter[#4]{#2}%
2671 \expandafter\string\csname\@glo@suffix\endcsname
2672 }%
2673 \@gls@checkmkidxchars#1%
2674 }
```

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.

```
2675 \def\@glo@check@mkidxrangechar#1#2\@nil{%
2676 \if#1(\relax
2677 \def\@glo@prefix{(%
2678 \if\relax#2\relax
2679 \def\@glo@suffix{glsnumberformat}%
2680 \else
2681 \def\@glo@suffix{#2}%
2682 \fi
2683 \else
2684 \if#1)\relax
2685 \def\@glo@prefix{)%
2686 \if\relax#2\relax
```

```

2687     \def\@glo@suffix{glsnumberformat}%
2688     \else
2689     \def\@glo@suffix{#2}%
2690 \fi
2691 \else
2692     \def\@glo@prefix{}\def\@glo@suffix{#1#2}%
2693 \fi
2694 \fi}

```

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

```

2695 \newcommand*{\@gls@escbsdq}[1]{%
2696   \def\@gls@checkedmkidx{}%
2697   \let\gls@xdystring=#1\relax
2698   \@onelevel@sanitize\gls@xdystring
2699   \edef\do@gls@xdycheckbackslash{%
2700     \noexpand\@gls@xdycheckbackslash\gls@xdystring\noexpand\@nil
2701     \@backslashchar\@backslashchar\noexpand\null}%
2702   \do@gls@xdycheckbackslash
2703   \expandafter\@gls@updatechecked\@gls@checkedmkidx{\gls@xdystring}%
2704   \def\@gls@checkedmkidx{}%
2705   \expandafter\@gls@xdycheckquote\gls@xdystring\@nil""\null
2706   \expandafter\@gls@updatechecked\@gls@checkedmkidx{\gls@xdystring}%

```

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

```

2707   \@for\@gls@tmp:=\gls@protected@pagefmts\do
2708   {%
2709     \edef\@gls@sanitized@tmp{\expandafter\@gobble\string\\expandonce\@gls@tmp}%
2710     \@onelevel@sanitize\@gls@sanitized@tmp
2711     \edef\gls@dostubst{%
2712       \noexpand\DTLsubstituteall\noexpand\gls@xdystring
2713       {\@gls@sanitized@tmp}{\expandonce\@gls@tmp}%
2714     }%
2715     \gls@dostubst
2716   }%

```

Assign to required control sequence

```

2717   \let#1=\gls@xdystring
2718 }

```

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

`\gls@checkmkidxchars`

```

2719 \newcommand{\@gls@checkmkidxchars}[1]{%
2720   \ifglxsindy
2721     \@gls@escbsdq{#1}%
2722   \else
2723     \def\@gls@checkedmkidx{}%
2724     \expandafter\@gls@checkquote#1\@nil""\null

```

```

2725 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2726 \def\@gls@checkedmkidx{}%
2727 \expandafter\@gls@checkescquote#1\@nil\""\null
2728 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2729 \def\@gls@checkedmkidx{}%
2730 \expandafter\@gls@checkescactual#1\@nil\?\?\null
2731 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2732 \def\@gls@checkedmkidx{}%
2733 \expandafter\@gls@checkactual#1\@nil??\null
2734 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2735 \def\@gls@checkedmkidx{}%
2736 \expandafter\@gls@checkbar#1\@nil||\null
2737 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2738 \def\@gls@checkedmkidx{}%
2739 \expandafter\@gls@checkesbar#1\@nil\\|\null
2740 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2741 \def\@gls@checkedmkidx{}%
2742 \expandafter\@gls@checklevel#1\@nil!!\null
2743 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2744 \fi
2745 }

```

Update the control sequence and strip trailing \@nil:

\@gls@updatechecked

```

2746 \def\@gls@updatechecked#1\@nil#2{\def#2{#1}}

```

\@gls@tmpb Define temporary token

```

2747 \newtoks\@gls@tmpb

```

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

```

2748 \def\@gls@checkquote#1"#2"#3\null{%
2749 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2750 \toks@={#1}%
2751 \ifx\null#2\null
2752 \ifx\null#3\null
2753 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2754 \def\@gls@checkquote{\relax}%
2755 \else
2756 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2757 \@gls@quotechar\@gls@quotechar\@gls@quotechar\@gls@quotechar}%
2758 \def\@gls@checkquote{\@gls@checkquote#3\null}%
2759 \fi
2760 \else
2761 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2762 \@gls@quotechar\@gls@quotechar}%
2763 \ifx\null#3\null
2764 \def\@gls@checkquote{\@gls@checkquote#2""\null}%
2765 \else

```

```

2766 \def\@gls@checkquote{\@gls@checkquote#2"#3\null}%
2767 \fi
2768 \fi
2769 \@gls@checkquote
2770 }

```

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

```

2771 \def\@gls@checkescquote#1\"#2\"#3\null{%
2772 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2773 \toks@={#1}%
2774 \ifx\null#2\null
2775 \ifx\null#3\null
2776 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2777 \def\@gls@checkescquote{\relax}%
2778 \else
2779 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2780 \@gls@quotechar\string\" \@gls@quotechar
2781 \@gls@quotechar\string\" \@gls@quotechar}%
2782 \def\@gls@checkescquote{\@gls@checkescquote#3\null}%
2783 \fi
2784 \else
2785 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2786 \@gls@quotechar\string\" \@gls@quotechar}%
2787 \ifx\null#3\null
2788 \def\@gls@checkescquote{\@gls@checkescquote#2\"\" \null}%
2789 \else
2790 \def\@gls@checkescquote{\@gls@checkescquote#2\"#3\null}%
2791 \fi
2792 \fi
2793 \@gls@checkescquote
2794 }

```

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

```

2795 \def\@gls@checkescactual#1\?#2\?#3\null{%
2796 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2797 \toks@={#1}%
2798 \ifx\null#2\null
2799 \ifx\null#3\null
2800 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2801 \def\@gls@checkescactual{\relax}%
2802 \else
2803 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2804 \@gls@quotechar\string\" \@gls@actualchar
2805 \@gls@quotechar\string\" \@gls@actualchar}%
2806 \def\@gls@checkescactual{\@gls@checkescactual#3\null}%
2807 \fi
2808 \else
2809 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2810 \@gls@quotechar\string\" \@gls@actualchar}%

```

```

2811 \ifx\null#3\null
2812 \def\@gls@checkescactual{\@gls@checkescactual#2\?\?\null}%
2813 \else
2814 \def\@gls@checkescactual{\@gls@checkescactual#2\?#3\null}%
2815 \fi
2816 \fi
2817 \@gls@checkescactual
2818 }

```

\@gls@checkescbar Similarly for \|:

```

2819 \def\@gls@checkescbar#1\|#2\|#3\null{%
2820 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2821 \toks@={#1}%
2822 \ifx\null#2\null
2823 \ifx\null#3\null
2824 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2825 \def\@gls@checkescbar{\relax}%
2826 \else
2827 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2828 \@gls@quotechar\string\\"\@gls@encapchar
2829 \@gls@quotechar\string\\"\@gls@encapchar}%
2830 \def\@gls@checkescbar{\@gls@checkescbar#3\null}%
2831 \fi
2832 \else
2833 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2834 \@gls@quotechar\string\\"\@gls@encapchar}%
2835 \ifx\null#3\null
2836 \def\@gls@checkescbar{\@gls@checkescbar#2\|\|\null}%
2837 \else
2838 \def\@gls@checkescbar{\@gls@checkescbar#2\|#3\null}%
2839 \fi
2840 \fi
2841 \@gls@checkescbar
2842 }

```

\@gls@checkesclevel Similarly for \!:

```

2843 \def\@gls@checkesclevel#1\!#2\!#3\null{%
2844 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2845 \toks@={#1}%
2846 \ifx\null#2\null
2847 \ifx\null#3\null
2848 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2849 \def\@gls@checkesclevel{\relax}%
2850 \else
2851 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2852 \@gls@quotechar\string\!\@gls@levelchar
2853 \@gls@quotechar\string\!\@gls@levelchar}%
2854 \def\@gls@checkesclevel{\@gls@checkesclevel#3\null}%
2855 \fi

```



```

2856 \else
2857 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2858 \@gls@quotechar\string\" \@gls@levelchar}%
2859 \ifx\@gls@checkedmkidx\@gls@checkedmkidx
2860 \def\@gls@checkedmkidx{\@gls@checkedmkidx#2\!\!\null}%
2861 \else
2862 \def\@gls@checkedmkidx{\@gls@checkedmkidx#2\!\!#3\null}%
2863 \fi
2864 \fi
2865 \@gls@checkedmkidx
2866 }

```

\@gls@checkbar and for |:

```

2867 \def\@gls@checkbar#1|#2|#3\null{%
2868 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2869 \toks@={#1}%
2870 \ifx\@gls@checkbar\@gls@checkbar
2871 \ifx\@gls@checkbar\@gls@checkbar
2872 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2873 \def\@gls@checkbar{\relax}%
2874 \else
2875 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2876 \@gls@quotechar\@gls@encapchar\@gls@quotechar\@gls@encapchar}%
2877 \def\@gls@checkbar{\@gls@checkbar#3\null}%
2878 \fi
2879 \else
2880 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2881 \@gls@quotechar\@gls@encapchar}%
2882 \ifx\@gls@checkbar\@gls@checkbar
2883 \def\@gls@checkbar{\@gls@checkbar#2||\null}%
2884 \else
2885 \def\@gls@checkbar{\@gls@checkbar#2|#3\null}%
2886 \fi
2887 \fi
2888 \@gls@checkbar
2889 }

```

\@gls@checklevel and for !:

```

2890 \def\@gls@checklevel#1!#2!#3\null{%
2891 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2892 \toks@={#1}%
2893 \ifx\@gls@checklevel\@gls@checklevel
2894 \ifx\@gls@checklevel\@gls@checklevel
2895 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2896 \def\@gls@checklevel{\relax}%
2897 \else
2898 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2899 \@gls@quotechar\@gls@levelchar\@gls@quotechar\@gls@levelchar}%
2900 \def\@gls@checklevel{\@gls@checklevel#3\null}%

```

```

2901 \fi
2902 \else
2903 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2904 \@gls@quotechar\@gls@levelchar}%
2905 \ifx\null#3\null
2906 \def\@gls@checklevel{\@gls@checklevel#2!!\null}%
2907 \else
2908 \def\@gls@checklevel{\@gls@checklevel#2!#3\null}%
2909 \fi
2910 \fi
2911 \@gls@checklevel
2912 }

```

\@gls@checkactual and for ?:

```

2913 \def\@gls@checkactual#1?#2?#3\null{%
2914 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2915 \toks@={#1}%
2916 \ifx\null#2\null
2917 \ifx\null#3\null
2918 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2919 \def\@gls@checkactual{\relax}%
2920 \else
2921 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2922 \@gls@quotechar\@gls@actualchar\@gls@quotechar\@gls@actualchar}%
2923 \def\@gls@checkactual{\@gls@checkactual#3\null}%
2924 \fi
2925 \else
2926 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2927 \@gls@quotechar\@gls@actualchar}%
2928 \ifx\null#3\null
2929 \def\@gls@checkactual{\@gls@checkactual#2??\null}%
2930 \else
2931 \def\@gls@checkactual{\@gls@checkactual#2?#3\null}%
2932 \fi
2933 \fi
2934 \@gls@checkactual
2935 }

```

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

```

2936 \def\@gls@xdycheckquote#1"#2"#3\null{%
2937 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2938 \toks@={#1}%
2939 \ifx\null#2\null
2940 \ifx\null#3\null
2941 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2942 \def\@gls@xdycheckquote{\relax}%
2943 \else
2944 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2945 \string"\string"}%

```

```

2946     \def\@gls@xdycheckquote{\@gls@xdycheckquote#3\null}%
2947     \fi
2948   \else
2949     \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2950       \string\}%
2951     \ifx\null#3\null
2952       \def\@gls@xdycheckquote{\@gls@xdycheckquote#2""\null}%
2953     \else
2954       \def\@gls@xdycheckquote{\@gls@xdycheckquote#2"#3\null}%
2955     \fi
2956   \fi
2957   \@gls@xdycheckquote
2958 }

```

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

```

\@gls@xdycheckbackslash
2959 \edef\def\@gls@xdycheckbackslash{%
2960   \noexpand\def\noexpand\@gls@xdycheckbackslash##1\@backslashchar
2961     ##2\@backslashchar##3\noexpand\null{%
2962     \noexpand\@gls@tmpb=\noexpand\expandafter
2963       {\noexpand\@gls@checkedmkidx}%
2964     \noexpand\toks@={##1}%
2965     \noexpand\ifx\noexpand\null##2\noexpand\null
2966       \noexpand\ifx\noexpand\null##3\noexpand\null
2967         \noexpand\edef\noexpand\@gls@checkedmkidx{%
2968           \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@}%
2969         \noexpand\def\noexpand\@gls@xdycheckbackslash{\relax}%
2970       \noexpand\else
2971         \noexpand\edef\noexpand\@gls@checkedmkidx{%
2972           \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
2973             \@backslashchar\@backslashchar\@backslashchar\@backslashchar}%
2974       \noexpand\def\noexpand\@gls@xdycheckbackslash{%
2975         \noexpand\@gls@xdycheckbackslash##3\noexpand\null}%
2976       \noexpand\fi
2977     \noexpand\else
2978       \noexpand\edef\noexpand\@gls@checkedmkidx{%
2979         \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
2980           \@backslashchar\@backslashchar}%
2981     \noexpand\ifx\noexpand\null##3\noexpand\null
2982       \noexpand\def\noexpand\@gls@xdycheckbackslash{%
2983         \noexpand\@gls@xdycheckbackslash##2\@backslashchar
2984           \@backslashchar\noexpand\null}%
2985     \noexpand\else
2986       \noexpand\def\noexpand\@gls@xdycheckbackslash{%
2987         \noexpand\@gls@xdycheckbackslash##2\@backslashchar
2988           ##3\noexpand\null}%
2989     \noexpand\fi
2990   \noexpand\fi
2991   \noexpand\@gls@xdycheckbackslash

```

```
2992 }%
```

```
2993 }
```

Now go ahead and define `\@gls@xdycheckbackslash`

```
2994 \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`.

```
2995 \ifcsundef{hyperlink}%
```

```
2996 {%
```

```
2997   \gdef\@glslink#1#2{#2}%
```

```
2998 }%
```

```
2999 {%
```

```
3000   \gdef\@glslink#1#2{\hyperlink{#1}{#2}}%
```

```
3001 }
```

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

```
3002 \newlength\gls@tmplen \ifcsundef{hypertarget}%
```

```
3003 {%
```

```
3004   \gdef\@glstarget#1#2{#2}%
```

```
3005 }%
```

```
3006 {%
```

```
3007   \gdef\@glstarget#1#2{%
```

```
3008     \settoheight{\gls@tmplen}{#2}%
```

```
3009     \raisebox{\gls@tmplen}{\hypertarget{#1}{}}#2%
```

```
3010   }%
```

```
3011 }
```

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

`\glsdisablehyper`

```
3012 \newcommand{\glsdisablehyper}{%
```

```
3013   \renewcommand*\@glslink[2]{##2}%
```

```
3014   \renewcommand*\@glstarget[2]{##2}%
```

```
3015 }
```

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

`\glsenablehyper`

```
3016 \newcommand{\glsenablehyper}{%
```

```
3017 \renewcommand*\@glslink[2]{\hyperlink{##1}{##2}}%
```

```
3018 \renewcommand*\@glstarget[2]{%
```

```
3019   \settoheight{\gls@tmplen}{##2}%
```

```
3020   \raisebox{\gls@tmplen}{\hypertarget{##1}{}}##2}}
```

Provide some convenience commands if not already defined:

```
3021 \providecommand{\@firstofthree}[3]{#1}
3022 \providecommand{\@secondofthree}[3]{#2}
3023 \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
3024 \newrobustcmd*{\gls}{\@ifstar\@sgls\@gls}
```

Define the starred form:

```
\@sgls
3025 \newcommand*{\@sgls}[1][\@gls[hyper=false,#1]]
```

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

```
\@gls
3026 \newcommand*{\@gls}[2][\@%
3027 \new@ifnextchar[\@gls@{#1}{#2}]{\@gls@{#1}{#2}[]}%
3028 }
```

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

```
3029 \def\@gls@#1#2[#3]{%
3030 \glsdoifexists{#2}%
3031 {%
3032 \edef\@glo@type{\glsentrytype{#2}}%

3033 \let\glsifplural\@secondoftwo
3034 \let\glsifscapscase\@firstofthree
3035 \let\glsifcustomtext\@empty
3036 \def\glsinsert{#3}%
```

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

```
3037 \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.

```

3038 \ifglsused{#2}%
3039 {%
3040 \@gls@link[#1]{#2}{\@glo@text}%
3041 }%
3042 {%
3043 \gls@checkisacronymlist\@glo@type
3044 \ifthenelse
3045 {(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote})}
3046 {OR \NOT\boolean{gls hyperfirst}}
3047 }%
3048 {%
3049 \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
3050 }%
3051 {%
3052 \@gls@link[#1]{#2}{\@glo@text}%
3053 }%
3054 }%

```

Indicate that this entry has now been used

```

3055 \ifKV@glslink@local
3056 \glslocalunset{#2}%
3057 \else
3058 \glsunset{#2}%
3059 \fi
3060 }%
3061 }

```

`\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`

```

3062 \newrobustcmd*{\Gls}{\@ifstar\@sGls\@Gls}

```

Define the starred form:

```

3063 \newcommand*{\@sGls}[1][\@Gls[hyper=false,#1]]

```

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

```

3064 \newcommand*{\@Gls}[2][\@Gls@{#1}{#2}]{\@Gls@{#1}{#2}[]}
3065 \new@ifnextchar{\@Gls@{#1}{#2}}{\@Gls@{#1}{#2}[]}
3066 }

```

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

```

3067 \def\@Gls@#1#2[#3]{%

```

```

3068 \glsdoifexists{#2}%
3069 {%
3070   \edef\@glo@type{\glsentrytype{#2}}%

3071   \let\glsifplural\@secondoftwo
3072   \let\glsifcaps\@secondofthree
3073   \let\glsifcustomtext\@empty
3074   \def\glsinsert{#3}%

```

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

```

3075   \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.

3076   \ifglsused{#2}%
3077   {%
3078     \@gls@link[#1]{#2}{\@glo@text}%
3079   }%
3080   {%
3081     \gls@checkisacronymlist\@glo@type
3082     \ifthenelse
3083     {%
3084       \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
3085       \OR \NOT\boolean{glshyperfirst}%
3086     }%
3087     {%
3088       \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
3089     }%
3090     {%
3091       \@gls@link[#1]{#2}{\@glo@text}%
3092     }%
3093   }%

```

Indicate that this entry has now been used

```

3094   \ifKV@glslink@local
3095   \glslocalunset{#2}%
3096   \else
3097   \glsunset{#2}%
3098   \fi
3099 }%
3100 }

```

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

\GLS

```

3101 \newrobustcmd*{\GLS}{\@ifstar\@sGLS\@GLS}

```

Define the starred form:

```

3102 \newcommand*{\@sGLS}[1][\@GLS[hyper=false,#1]]{

```

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

```

3103 \newcommand*{\@GLS}[2][\@GLS@#1]{%
3104   \new@ifnextchar[\@GLS@#1]{\@GLS@#1}{\@GLS@#1}{\@GLS@#1}}%
3105 }

```

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

```

3106 \def\@GLS@#1#2[#3]{%
3107   \glsdoifexists{#2}%
3108   {%
3109     \edef\@glo@type{\glsentrytype{#2}}%
3110     \let\glsifplural\@secondoftwo
3111     \let\glsupcase\@thirdofthree
3112     \let\glscustomtext\@empty
3113     \def\glsinsert{#3}%

```

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

```

3114   \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.

```

3115   \ifglsused{#2}%
3116   {%
3117     \@gls@link[#1]{#2}{\@glo@text}%
3118   }%
3119   {%
3120     \gls@checkisacronymlist\@glo@type
3121     \ifthenelse
3122     {%
3123       \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
3124       \OR \NOT\boolean{glshyperfirst}}{%
3125       \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
3126     }%
3127     {%
3128       \@gls@link[#1]{#2}{\@glo@text}%
3129     }%
3130   }%

```

Indicate that this entry has now been used

```

3131   \ifKV@glslink@local
3132   \glslocalunset{#2}%
3133   \else
3134   \glsunset{#2}%
3135   \fi
3136 }%
3137 }

```

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



`\glspl`

```
3138 \newrobustcmd*{\glspl}{\@ifstar\sglspl\glspl}
```

Define the starred form:

```
3139 \newcommand*{\sglspl}[1][\@glspl[hyper=false,#1]]
```

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

```
3140 \newcommand*{\glspl}[2][\@glspl]
```

```
3141 \new@ifnextchar[\@glspl@{#1}{#2}]{\@glspl@{#1}{#2}}%
```

```
3142 }
```

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

```
3143 \def\@glspl@#1#2[#3]{%
```

```
3144 \glsdoifexists{#2}%
```

```
3145 {%
```

```
3146 \edef\@glo@type{\glsentrytype{#2}}%
```

```
3147 \let\glsifplural\@firstoftwo
```

```
3148 \let\glsapscase\@firstofthree
```

```
3149 \let\glsapscustomtext\@empty
```

```
3150 \def\glsinsert{#3}%
```

```
3151 % Determine what the link text should be (this is stored in
```

```
3152 % \cs{@glo@text})
```

```
3153 %\changes{1.12}{2008 Mar 8}{now uses \cs{glsentrydescplural} and
```

```
3154 % \cs{glsentrysymbolplural} instead of \cs{glsentrydesc} and
```

```
3155 % \cs{glsentrysymbol}}
```

```
3156 %\changes{3.11a}{2013-10-15}{change to using \cs{glsentryfmt} style
```

```
3157 %commands}
```

```
3158 % \begin{macrocode}
```

```
3159 \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.

```
3160 \ifglsused{#2}%
```

```
3161 {%
```

```
3162 \@gls@link[#1]{#2}{\@glo@text}%
```

```
3163 }%
```

```
3164 {%
```

```
3165 \gls@checkisacronymlist\@glo@type
```

```
3166 \ifthenelse
```

```
3167 {%
```

```
3168 \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote})\
```

```
3169 \OR \NOT\boolean{glshyperfirst}}%
```

```
3170 }%
```

```
3171 {%
```

```
3172 \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
```

```
3173 }%
```

```
3174 {%
```

```

3175      \@gls@link[#1]{#2}{\@glo@text}%
3176    }%
3177  }%

```

Indicate that this entry has now been used

```

3178    \ifKV@glslink@local
3179      \glslocalunset{#2}%
3180    \else
3181      \glsunset{#2}%
3182    \fi
3183  }%
3184 }

```

`\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`

```

3185 \newrobustcmd*{\Glspl}{\@ifstar\@sGlspl\@Glspl}

```

Define the starred form:

```

3186 \newcommand*{\@sGlspl}[1][\@Glspl[hyper=false,#1]]{

```

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

```

3187 \newcommand*{\@Glspl}[2][\@Glspl@{#1}{#2}]{\@Glspl@{#1}{#2}[1]}%
3188 \new@ifnextchar[\@Glspl@{#1}{#2}]{\@Glspl@{#1}{#2}[1]}%
3189 }

```

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

```

3190 \def\@Glspl@#1#2[#3]{%
3191   \glsdoifexists{#2}%
3192   {%
3193     \edef\@glo@type{\glsentrytype{#2}}%
3194     \let\glsifplural\@firstoftwo
3195     \let\glscapscase\@secondofthree
3196     \let\glscustomtext\@empty
3197     \def\glsinsert{#3}%

```

Determine what the link text should be (this is stored in `\@glo@text`). This needs to be expanded so that the `\@glo@text` can be passed to `\xmakefirstuc`.

```

3198   \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.

```

3199   \ifglused{#2}%
3200   {%

```

```

3201      \@gls@link[#1]{#2}{\@glo@text}%
3202    }%
3203    {%
3204      \gls@checkisacronymlist\@glo@type
3205      \ifthenelse
3206        {%
3207          \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
3208          \OR \NOT\boolean{glshyperfirst}}%
3209        }%
3210        {%
3211          \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
3212        }%
3213        {%
3214          \@gls@link[#1]{#2}{\@glo@text}%
3215        }%
3216      }%

```

Indicate that this entry has now been used

```

3217      \ifKV@glslink@local
3218      \glslocalunset{#2}%
3219      \else
3220      \glsunset{#2}%
3221      \fi
3222    }%
3223 }

```

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

\GLSp1

```

3224 \newrobustcmd*{\GLSp1}{\@ifstar\@sGLSp1\@GLSp1}

```

Define the starred form:

```

3225 \newcommand*{\@sGLSp1}[1] [] {\@GLSp1[hyper=false,#1]}

```

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

```

3226 \newcommand*{\@GLSp1}[2] [] {%
3227   \new@ifnextchar[\@GLSp1@{#1}{#2}]{\@GLSp1@{#1}{#2} []}%
3228 }

```

\@GLSp1 Read in the final optional argument:

```

3229 \def\@GLSp1@#1#2[#3]{%
3230   \glsdoifexists{#2}%
3231   {%
3232     \edef\@glo@type{\glsentrytype{#2}}%
3233     \let\glsifplural\@firstoftwo
3234     \let\glsifscapscase\@thirdofthree
3235     \let\glsifcustomtext\@empty
3236     \def\glsinsert{#3}%

```

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

```

3237 \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.
3238 \ifglsused{#2}%
3239 {%
3240 \@gls@link[#1]{#2}{\@glo@text}%
3241 }%
3242 {%
3243 \gls@checkisacronymlist\@glo@type
3244 \ifthenelse
3245 {%
3246 \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
3247 \OR \NOT\boolean{glshyperfirst}%
3248 }%
3249 {%
3250 \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
3251 }%
3252 {%
3253 \@gls@link[#1]{#2}{\@glo@text}%
3254 }%
3255 }%

```

Indicate that this entry has now been used

```

3256 \ifKV@glslink@local
3257 \glslocalunset{#2}%
3258 \else
3259 \glsunset{#2}%
3260 \fi
3261 }%
3262 }

```

`\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:

```

3263 \newrobustcmd*{\glsdisp}{\@ifstar\@sglsdisp\@glsdisp}

```

Define the starred form:

```

\@sgls
3264 \newcommand*{\@sglsdisp}[1] [] {\@glsdisp[hyper=false,#1]}

```

Defined the un-starred form.

```

\@glsdisp
3265 \newcommand*{\@glsdisp}[3] [] {%
3266 \glsdoifexists{#2}{%

```

```
3267 \edef\@glo@type{\glsentrytype{#2}}%
```

```
3268 \let\glsifplural\@secondoftwo
```

```
3269 \let\glsupcase\@firstofthree
```

```
3270 \def\glscustomtext{#3}%
```

```
3271 \def\glsinsert{}%
```

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

```
3272 \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.

```
3273 \ifglsused{#2}%
```

```
3274 {%
```

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

```
3276 }%
```

```
3277 {%
```

```
3278 \gls@checkisacronymlist\@glo@type
```

```
3279 \ifthenelse{\boolean{@glsisacronymlist}}\AND
```

```
3280 \boolean{glsacrfootnote}) \OR \NOT\boolean{glshyperfirst}}%
```

```
3281 {%
```

```
3282 \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
```

```
3283 }%
```

```
3284 {%
```

```
3285 \@gls@link[#1]{#2}{\@glo@text}%
```

```
3286 }%
```

```
3287 }%
```

Indicate that this entry has now been used

```
3288 \ifKV@glslink@local
```

```
3289 \glslocalunset{#2}%
```

```
3290 \else
```

```
3291 \glsunset{#2}%
```

```
3292 \fi
```

```
3293 }%
```

```
3294 }
```

\@gls@field@link

```
3295 \newcommand{\@gls@field@link}[3]{%
```

```
3296 \glsdoifexists{#2}%
```

```
3297 {%
```

```
3298 \edef\@glo@type{\glsentrytype{#2}}%
```

```
3299 \@gls@link[#1]{#2}{#3}%
```

```
3300 }%
```

```
3301 }
```

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

`\glstext`

```
3302 \newrobustcmd*{\glstext}{\@ifstar\@sglstext\@glstext}
```

Define the starred form:

```
3303 \newcommand*{\@sglstext}[1] [] {\@glstext[hyper=false,#1]}
```

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

```
3304 \newcommand*{\@glstext}[2] [] {%
```

```
3305   \new@ifnextchar[{\@glstext@{#1}{#2}}{\@glstext@{#1}{#2} []}]
```

Read in the final optional argument:

```
3306 \def\@glstext@#1#2[#3] {%
```

```
3307   \@gls@field@link{#1}{#2}{\glsentrytext{#2}#3}%
```

```
3308 }
```

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

`\GLStext`

```
3309 \newrobustcmd*{\GLStext}{\@ifstar\@sGLStext\@GLStext}
```

Define the starred form:

```
3310 \newcommand*{\@sGLStext}[1] [] {\@GLStext[hyper=false,#1]}
```

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

```
3311 \newcommand*{\@GLStext}[2] [] {%
```

```
3312   \new@ifnextchar[{\@GLStext@{#1}{#2}}{\@GLStext@{#1}{#2} []}]
```

Read in the final optional argument:

```
3313 \def\@GLStext@#1#2[#3] {%
```

```
3314   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentrytext{#2}#3}}%
```

```
3315 }
```

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

`\Glstext`

```
3316 \newrobustcmd*{\Glstext}{\@ifstar\@sGlstext\@Glstext}
```

Define the starred form:

```
3317 \newcommand*{\@sGlstext}[1] [] {\@Glstext[hyper=false,#1]}
```

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

```
3318 \newcommand*{\@Glstext}[2] [] {%
```

```
3319   \new@ifnextchar[{\@Glstext@{#1}{#2}}{\@Glstext@{#1}{#2} []}]
```

Read in the final optional argument:

```
3320 \def\@Glstext@#1#2[#3] {%
```

```
3321   \@gls@field@link{#1}{#2}{\Glsentrytext{#2}#3}%
```

```
3322 }
```

`\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`

```
3323 \newrobustcmd*{\glsfirst}{\@ifstar\sglsfirst\glsfirst}
```

Define the starred form:

```
3324 \newcommand*{\sglsfirst}[1] [] {\glsfirst[hyper=false,#1]}
```

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

```
3325 \newcommand*{\glsfirst}[2] [] {%
```

```
3326   \new@ifnextchar[{\glsfirst@{#1}{#2}}{\glsfirst@{#1}{#2} []}]
```

Read in the final optional argument:

```
3327 \def\glsfirst@#1#2[#3] {%
```

```
3328   \gls@field@link{#1}{#2}{\glsentryfirst{#2}#3}%
```

```
3329 }
```

`\Glsfirst` behaves like `\glsfirst` except it displays the first letter in uppercase.

`\Glsfirst`

```
3330 \newrobustcmd*{\Glsfirst}{\@ifstar\sglsfirst\Glsfirst}
```

Define the starred form:

```
3331 \newcommand*{\sglsfirst}[1] [] {\Glsfirst[hyper=false,#1]}
```

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

```
3332 \newcommand*{\Glsfirst}[2] [] {%
```

```
3333   \new@ifnextchar[{\Glsfirst@{#1}{#2}}{\Glsfirst@{#1}{#2} []}]
```

Read in the final optional argument:

```
3334 \def\Glsfirst@#1#2[#3] {%
```

```
3335   \gls@field@link{#1}{#2}{\Glsentryfirst{#2}#3}%
```

```
3336 }
```

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

`\GLSfirst`

```
3337 \newrobustcmd*{\GLSfirst}{\@ifstar\sglsfirst\GLSfirst}
```

Define the starred form:

```
3338 \newcommand*{\sglsfirst}[1] [] {\GLSfirst[hyper=false,#1]}
```

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

```
3339 \newcommand*{\GLSfirst}[2] [] {%
```

```
3340   \new@ifnextchar[{\GLSfirst@{#1}{#2}}{\GLSfirst@{#1}{#2} []}]
```

Read in the final optional argument:

```
3341 \def\@GLSfirst@#1#2[#3]{%
3342   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glstryfirst{#2}#3}}%
3343 }
```

`\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`

```
3344 \newrobustcmd*{\glsplural}{\@ifstar\@sglsplural\@glsplural}
```

Define the starred form:

```
3345 \newcommand*{\@sglsplural}[1][]{\@glsplural[hyper=false,#1]}
```

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

```
3346 \newcommand*{\@glsplural}[2][]{%
3347   \new@ifnextchar[{\@glsplural@{#1}{#2}}{\@glsplural@{#1}{#2}[]}]}
```

Read in the final optional argument:

```
3348 \def\@glsplural@#1#2[#3]{%
3349   \@gls@field@link{#1}{#2}{\glstryplural{#2}#3}%
3350 }
```

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

`\Glsplural`

```
3351 \newrobustcmd*{\Glsplural}{\@ifstar\@sGlsplural\@Glsplural}
```

Define the starred form:

```
3352 \newcommand*{\@sGlsplural}[1][]{\@Glsplural[hyper=false,#1]}
```

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

```
3353 \newcommand*{\@Glsplural}[2][]{%
3354   \new@ifnextchar[{\@Glsplural@{#1}{#2}}{\@Glsplural@{#1}{#2}[]}]}
```

Read in the final optional argument:

```
3355 \def\@Glsplural@#1#2[#3]{%
3356   \@gls@field@link{#1}{#2}{\Glsentryplural{#2}#3}%
3357 }
```

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

`\GLSplural`

```
3358 \newrobustcmd*{\GLSplural}{\@ifstar\@sGLSplural\@GLSplural}
```

Define the starred form:

```
3359 \newcommand*{\@sGLSplural}[1][]{\@GLSplural[hyper=false,#1]}
```



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

```
3360 \newcommand*{\@GLSplural}[2] [] {%
3361   \new@ifnextchar[{\@GLSplural@{#1}{#2}}{\@GLSplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3362 \def\@GLSplural@#1#2[#3] {%
3363   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryplural{#2}#3}}%
3364 }
```

`\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`

```
3365 \newrobustcmd*{\glsfirstplural}{\@ifstar\@sglsfirstplural\@glsfirstplural}
```

Define the starred form:

```
3366 \newcommand*{\@sglsfirstplural}[1] [] {\@glsfirstplural[hyper=false,#1]}
```

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

```
3367 \newcommand*{\@glsfirstplural}[2] [] {%
3368   \new@ifnextchar[{\@glsfirstplural@{#1}{#2}}{\@glsfirstplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3369 \def\@glsfirstplural@#1#2[#3] {%
3370   \@gls@field@link{#1}{#2}{\glsentryfirstplural{#2}#3}%
3371 }
```

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

`\Glsfirstplural`

```
3372 \newrobustcmd*{\Glsfirstplural}{\@ifstar\@sGlsfirstplural\@Glsfirstplural}
```

Define the starred form:

```
3373 \newcommand*{\@sGlsfirstplural}[1] [] {\@Glsfirstplural[hyper=false,#1]}
```

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

```
3374 \newcommand*{\@Glsfirstplural}[2] [] {%
3375   \new@ifnextchar[{\@Glsfirstplural@{#1}{#2}}{\@Glsfirstplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3376 \def\@Glsfirstplural@#1#2[#3] {%
3377   \@gls@field@link{#1}{#2}{\Glsentryfirstplural{#2}#3}%
3378 }
```

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

`\GLSfirstplural`

```
3379 \newrobustcmd*{\GLSfirstplural}{\@ifstar\@sGLSfirstplural\@GLSfirstplural}
```

Define the starred form:

```
3380 \newcommand*{\@sGLSfirstplural}[1] [] {\@GLSfirstplural[hyper=false,#1]}
```

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

```
3381 \newcommand*{\@GLSfirstplural}[2] [] {%
```

```
3382 \new@ifnextchar[\@GLSfirstplural@{#1}{#2}]{\@GLSfirstplural@{#1}{#2} []}}
```

Read in the final optional argument:

```
3383 \def\@GLSfirstplural@#1#2[#3] {%
```

```
3384 \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryfirstplural{#2}#3}}%
```

```
3385 }
```

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

`\glsname`

```
3386 \newrobustcmd*{\glsname}{\@ifstar\@sglsname\@glsname}
```

Define the starred form:

```
3387 \newcommand*{\@sglsname}[1] [] {\@glsname[hyper=false,#1]}
```

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

```
3388 \newcommand*{\@glsname}[2] [] {%
```

```
3389 \new@ifnextchar[\@glsname@{#1}{#2}]{\@glsname@{#1}{#2} []}}
```

Read in the final optional argument:

```
3390 \def\@glsname@#1#2[#3] {%
```

```
3391 \@gls@field@link{#1}{#2}{\glsentryname{#2}#3}%
```

```
3392 }
```

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

`\Glsname`

```
3393 \newrobustcmd*{\Glsname}{\@ifstar\@sGlsname\@Glsname}
```

Define the starred form:

```
3394 \newcommand*{\@sGlsname}[1] [] {\@Glsname[hyper=false,#1]}
```

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

```
3395 \newcommand*{\@Glsname}[2] [] {%
```

```
3396 \new@ifnextchar[\@Glsname@{#1}{#2}]{\@Glsname@{#1}{#2} []}}
```

Read in the final optional argument:

```
3397 \def\@Glsname@#1#2[#3] {%
```

```
3398 \@gls@field@link{#1}{#2}{\Glsentryname{#2}#3}%
```

```
3399 }
```

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

`\GLSname`

```
3400 \newrobustcmd*{\GLSname}{\@ifstar\@sGLSname\@GLSname}
```

Define the starred form:

```
3401 \newcommand*{\@sGLSname}[1] [] {\@GLSname[hyper=false,#1]}
```

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

```
3402 \newcommand*{\@GLSname}[2] [] {%
```

```
3403   \new@ifnextchar[{\@GLSname@{#1}{#2}}{\@GLSname@{#1}{#2} []}]
```

Read in the final optional argument:

```
3404 \def\@GLSname@#1#2[#3] {%
```

```
3405   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryname{#2}#3}}%
```

```
3406 }
```

`\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`

```
3407 \newrobustcmd*{\glsdesc}{\@ifstar\@sglsdesc\@glsdesc}
```

Define the starred form:

```
3408 \newcommand*{\@sglsdesc}[1] [] {\@glsdesc[hyper=false,#1]}
```

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

```
3409 \newcommand*{\@glsdesc}[2] [] {%
```

```
3410   \new@ifnextchar[{\@glsdesc@{#1}{#2}}{\@glsdesc@{#1}{#2} []}]
```

Read in the final optional argument:

```
3411 \def\@glsdesc@#1#2[#3] {%
```

```
3412   \@gls@field@link{#1}{#2}{\glsentrydesc{#2}#3}%
```

```
3413 }
```

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

`\Glsdesc`

```
3414 \newrobustcmd*{\Glsdesc}{\@ifstar\@sGlsdesc\@Glsdesc}
```

Define the starred form:

```
3415 \newcommand*{\@sGlsdesc}[1] [] {\@Glsdesc[hyper=false,#1]}
```

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

```
3416 \newcommand*{\@Glsdesc}[2] [] {%
```

```
3417   \new@ifnextchar[{\@Glsdesc@{#1}{#2}}{\@Glsdesc@{#1}{#2} []}]
```

Read in the final optional argument:

```
3418 \def\@Glsdesc@#1#2[#3] {%
```

```
3419   \@gls@field@link{#1}{#2}{\Glsentrydesc{#2}#3}%
```

```
3420 }
```

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

`\GLSdesc`

```
3421 \newrobustcmd*{\GLSdesc}{\@ifstar\@sGLSdesc\@GLSdesc}
```

Define the starred form:

```
3422 \newcommand*{\@sGLSdesc}[1] [] {\@GLSdesc[hyper=false,#1]}
```

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

```
3423 \newcommand*{\@GLSdesc}[2] [] {%
```

```
3424   \new@ifnextchar[{\@GLSdesc@{#1}{#2}}{\@GLSdesc@{#1}{#2} []}]
```

Read in the final optional argument:

```
3425 \def\@GLSdesc@#1#2[#3] {%
```

```
3426   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentrydesc{#2}#3}}%
```

```
3427 }
```

`\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`

```
3428 \newrobustcmd*{\glsdescplural}{\@ifstar\@sglsdescplural\@glsdescplural}
```

Define the starred form:

```
3429 \newcommand*{\@sglsdescplural}[1] [] {\@glsdescplural[hyper=false,#1]}
```

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

```
3430 \newcommand*{\@glsdescplural}[2] [] {%
```

```
3431   \new@ifnextchar[{\@glsdescplural@{#1}{#2}}{\@glsdescplural@{#1}{#2} []}]
```

Read in the final optional argument:

```
3432 \def\@glsdescplural@#1#2[#3] {%
```

```
3433   \@gls@field@link{#1}{#2}{\glsentrydescplural{#2}#3}}%
```

```
3434 }
```

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

`\Glsdescplural`

```
3435 \newrobustcmd*{\Glsdescplural}{\@ifstar\@sGlsdescplural\@Glsdescplural}
```

Define the starred form:

```
3436 \newcommand*{\@sGlsdescplural}[1] [] {\@Glsdescplural[hyper=false,#1]}
```

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

```
3437 \newcommand*{\@Glsdescplural}[2] [] {%
```

```
3438   \new@ifnextchar[{\@Glsdescplural@{#1}{#2}}{\@Glsdescplural@{#1}{#2} []}]
```

Read in the final optional argument:

```
3439 \def\@GLSdescplural@#1#2[#3]{%
3440   \@gls@field@link{#1}{#2}{\Glsentrydescplural{#2}#3}%
3441 }
```

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

`\GLSdescplural`

```
3442 \newrobustcmd*{\GLSdescplural}{\@ifstar\@sGLSdescplural\@GLSdescplural}
```

Define the starred form:

```
3443 \newcommand*{\@sGLSdescplural}[1] [] {\@GLSdescplural[hyper=false,#1]}
```

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

```
3444 \newcommand*{\@GLSdescplural}[2] [] {%
3445   \new@ifnextchar[\@GLSdescplural@{#1}{#2}]{\@GLSdescplural@{#1}{#2} []}}
```

Read in the final optional argument:

```
3446 \def\@GLSdescplural@#1#2[#3]{%
3447   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\Glsentrydescplural{#2}#3}}%
3448 }
```

`\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`

```
3449 \newrobustcmd*{\glsymbol}{\@ifstar\@sglsymbol\@glsymbol}
```

Define the starred form:

```
3450 \newcommand*{\@sglsymbol}[1] [] {\@glsymbol[hyper=false,#1]}
```

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

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

Read in the final optional argument:

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

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

`\Glsymbol`

```
3456 \newrobustcmd*{\Glsymbol}{\@ifstar\@sGlsymbol\@Glsymbol}
```

Define the starred form:

```
3457 \newcommand*{\@sGlsymbol}[1] [] {\@Glsymbol[hyper=false,#1]}
```

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

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

Read in the final optional argument:

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

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

\GLSsymbol

```
3463 \newrobustcmd*{\GLSsymbol}{\@ifstar\@sGLSsymbol\@GLSsymbol}
```

Define the starred form:

```
3464 \newcommand*{\@sGLSsymbol}[1] [] {\@GLSsymbol[hyper=false,#1]}
```

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

```
3465 \newcommand*{\@GLSsymbol}[2] [] {%
3466   \new@ifnextchar[{\@GLSsymbol@{#1}{#2}}{\@GLSsymbol@{#1}{#2} [] }}
```

Read in the final optional argument:

```
3467 \def\@GLSsymbol@#1#2[#3] {%
3468   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glstentrysymbol{#2}#3}}%
3469 }
```

\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

```
3470 \newrobustcmd*{\glsymbolplural}{\@ifstar\@sglsymbolplural\@glsymbolplural}
```

Define the starred form:

```
3471 \newcommand*{\@sglsymbolplural}[1] [] {\@glsymbolplural[hyper=false,#1]}
```

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

```
3472 \newcommand*{\@glsymbolplural}[2] [] {%
3473   \new@ifnextchar[{\@glsymbolplural@{#1}{#2}}{\@glsymbolplural@{#1}{#2} [] }}
```

Read in the final optional argument:

```
3474 \def\@glsymbolplural@#1#2[#3] {%
3475   \@gls@field@link{#1}{#2}{\glstentrysymbolplural{#2}#3}%
3476 }
```

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

\Glsymbolplural

```
3477 \newrobustcmd*{\Glsymbolplural}{\@ifstar\@sGlsymbolplural\@Glsymbolplural}
```

Define the starred form:

```
3478 \newcommand*{\@sGlssymbolplural}[1] [] {\@Glssymbolplural[hyper=false,#1]}
```

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

```
3479 \newcommand*{\@Glssymbolplural}[2] [] {%
```

```
3480   \new@ifnextchar[{\@Glssymbolplural@{#1}{#2}}{\@Glssymbolplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3481 \def\@Glssymbolplural@#1#2[#3] {%
```

```
3482   \@gls@field@link{#1}{#2}{\glstentrysymbolplural{#2}#3}%
```

```
3483 }
```

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

`\GLSsymbolplural`

```
3484 \newrobustcmd*{\GLSsymbolplural}{\@ifstar\@sGLSsymbolplural\@GLSsymbolplural}
```

Define the starred form:

```
3485 \newcommand*{\@sGLSsymbolplural}[1] [] {\@GLSsymbolplural[hyper=false,#1]}
```

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

```
3486 \newcommand*{\@GLSsymbolplural}[2] [] {%
```

```
3487   \new@ifnextchar[{\@GLSsymbolplural@{#1}{#2}}{\@GLSsymbolplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3488 \def\@GLSsymbolplural@#1#2[#3] {%
```

```
3489   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glstentrysymbolplural{#2}#3}}%
```

```
3490 }
```

`\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`

```
3491 \newrobustcmd*{\glsuseri}{\@ifstar\@sglsuseri\@glsuseri}
```

Define the starred form:

```
3492 \newcommand*{\@sglsuseri}[1] [] {\@glsuseri[hyper=false,#1]}
```

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

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

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

Read in the final optional argument:

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

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

```
3497 }
```

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

`\Glsuseri`

```
3498 \newrobustcmd*{\Glsuseri}{\@ifstar\@sGlsuseri\@Glsuseri}
```

Define the starred form:

```
3499 \newcommand*{\@sGlsuseri}[1] [] {\@Glsuseri[hyper=false,#1]}
```

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

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

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

Read in the final optional argument:

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

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

```
3504 }
```

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

`\GLSuseri`

```
3505 \newrobustcmd*{\GLSuseri}{\@ifstar\@sGLSuseri\@GLSuseri}
```

Define the starred form:

```
3506 \newcommand*{\@sGLSuseri}[1] [] {\@GLSuseri[hyper=false,#1]}
```

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

```
3507 \newcommand*{\@GLSuseri}[2] [] {%
```

```
3508   \new@ifnextchar[{\@GLSuseri@{#1}{#2}}{\@GLSuseri@{#1}{#2} []}]
```

Read in the final optional argument:

```
3509 \def\@GLSuseri@#1#2[#3] {%
```

```
3510   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryuseri{#2}#3}}%
```

```
3511 }
```

`\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`

```
3512 \newrobustcmd*{\glsuserii}{\@ifstar\@sglsuserii\@glsuserii}
```

Define the starred form:

```
3513 \newcommand*{\@sglsuserii}[1] [] {\@glsuserii[hyper=false,#1]}
```

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

```
3514 \newcommand*{\@glsuserii}[2] [] {%
```

```
3515   \new@ifnextchar[{\@glsuserii@{#1}{#2}}{\@glsuserii@{#1}{#2} []}]
```

Read in the final optional argument:

```
3516 \def\@glsuserii@#1#2[#3] {%
```

```
3517   \@gls@field@link{#1}{#2}{\glsentryuserii{#2}#3}%
```

```
3518 }
```



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

`\Glsuserii`

```
3519 \newrobustcmd*{\Glsuserii}{\@ifstar\@sGlsuserii\@Glsuserii}
```

Define the starred form:

```
3520 \newcommand*{\@sGlsuserii}[1] [] {\@Glsuserii[hyper=false,#1]}
```

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

```
3521 \newcommand*{\@Glsuserii}[2] [] {%
```

```
3522   \new@ifnextchar[{\@Glsuserii@{#1}{#2}}{\@Glsuserii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3523 \def\@Glsuserii@#1#2[#3] {%
```

```
3524   \@gls@field@link{#1}{#2}{\Glsentryuserii{#2}#3}%
```

```
3525 }
```

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

`\GLSuserii`

```
3526 \newrobustcmd*{\GLSuserii}{\@ifstar\@sGLSuserii\@GLSuserii}
```

Define the starred form:

```
3527 \newcommand*{\@sGLSuserii}[1] [] {\@GLSuserii[hyper=false,#1]}
```

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

```
3528 \newcommand*{\@GLSuserii}[2] [] {%
```

```
3529   \new@ifnextchar[{\@GLSuserii@{#1}{#2}}{\@GLSuserii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3530 \def\@GLSuserii@#1#2[#3] {%
```

```
3531   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\Glsentryuserii{#2}#3}}%
```

```
3532 }
```

`\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`

```
3533 \newrobustcmd*{\glsuseriii}{\@ifstar\@sglsuseriii\@glsuseriii}
```

Define the starred form:

```
3534 \newcommand*{\@sglsuseriii}[1] [] {\@glsuseriii[hyper=false,#1]}
```

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

```
3535 \newcommand*{\@glsuseriii}[2] [] {%
```

```
3536   \new@ifnextchar[{\@glsuseriii@{#1}{#2}}{\@glsuseriii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3537 \def\@glsuseriii@#1#2[#3]{%
3538   \@gls@field@link{#1}{#2}{\glstentryuseriii{#2}#3}%
3539 }
```

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

\Glsuseriii

```
3540 \newrobustcmd*{\Glsuseriii}{\@ifstar\@sGlsuseriii\@Glsuseriii}
```

Define the starred form:

```
3541 \newcommand*{\@sGlsuseriii}[1] [] {\@Glsuseriii[hyper=false,#1]}
```

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

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

Read in the final optional argument:

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

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

\GLSuseriii

```
3547 \newrobustcmd*{\GLSuseriii}{\@ifstar\@sGLSuseriii\@GLSuseriii}
```

Define the starred form:

```
3548 \newcommand*{\@sGLSuseriii}[1] [] {\@GLSuseriii[hyper=false,#1]}
```

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

```
3549 \newcommand*{\@GLSuseriii}[2] [] {%
3550   \new@ifnextchar[{\@GLSuseriii@{#1}{#2}}{\@GLSuseriii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3551 \def\@GLSuseriii@#1#2[#3]{%
3552   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glstentryuseriii{#2}#3}}%
3553 }
```

\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

```
3554 \newrobustcmd*{\glsuseriv}{\@ifstar\@sglsuseriv\@glsuseriv}
```

Define the starred form:

```
3555 \newcommand*{\@sglsuseriv}[1] [] {\@glsuseriv[hyper=false,#1]}
```

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

```
3556 \newcommand*{\@glsuseriv}[2] [] {%
3557   \new@ifnextchar[{\@glsuseriv@{#1}{#2}}{\@glsuseriv@{#1}{#2} [] }}
```

Read in the final optional argument:

```
3558 \def\@glsuseriv@#1#2[#3] {%
3559   \@gls@field@link{#1}{#2}{\glsentryuseriv{#2}#3}%
3560 }
```

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

\Glsuseriv

```
3561 \newrobustcmd*{\Glsuseriv}{\@ifstar\@sGlsuseriv\@Glsuseriv}
```

Define the starred form:

```
3562 \newcommand*{\@sGlsuseriv}[1] [] {\@Glsuseriv[hyper=false,#1]}
```

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

```
3563 \newcommand*{\@Glsuseriv}[2] [] {%
3564   \new@ifnextchar[{\@Glsuseriv@{#1}{#2}}{\@Glsuseriv@{#1}{#2} [] }}
```

Read in the final optional argument:

```
3565 \def\@Glsuseriv@#1#2[#3] {%
3566   \@gls@field@link{#1}{#2}{\Glsentryuseriv{#2}#3}%
3567 }
```

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

\GLSuseriv

```
3568 \newrobustcmd*{\GLSuseriv}{\@ifstar\@sGLSuseriv\@GLSuseriv}
```

Define the starred form:

```
3569 \newcommand*{\@sGLSuseriv}[1] [] {\@GLSuseriv[hyper=false,#1]}
```

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

```
3570 \newcommand*{\@GLSuseriv}[2] [] {%
3571   \new@ifnextchar[{\@GLSuseriv@{#1}{#2}}{\@GLSuseriv@{#1}{#2} [] }}
```

Read in the final optional argument:

```
3572 \def\@GLSuseriv@#1#2[#3] {%
3573   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryuseriv{#2}#3}}%
3574 }
```

\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

```
3575 \newrobustcmd*{\glsuserv}{\@ifstar\@sglsuserv\@glsuserv}
```

Define the starred form:

```
3576 \newcommand*{\@sglsuserv}[1] [] {\@glsuserv[hyper=false,#1]}
```

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

```
3577 \newcommand*{\@glsuserv}[2] [] {%
```

```
3578   \new@ifnextchar[{\@glsuserv@{#1}{#2}}{\@glsuserv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3579 \def\@glsuserv@#1#2[#3] {%
```

```
3580   \@gls@field@link{#1}{#2}{\glsentryuserv{#2}#3}%
```

```
3581 }
```

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

\Glsuserv

```
3582 \newrobustcmd*{\Glsuserv}{\@ifstar\@sGlsuserv\@Glsuserv}
```

Define the starred form:

```
3583 \newcommand*{\@sGlsuserv}[1] [] {\@Glsuserv[hyper=false,#1]}
```

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

```
3584 \newcommand*{\@Glsuserv}[2] [] {%
```

```
3585   \new@ifnextchar[{\@Glsuserv@{#1}{#2}}{\@Glsuserv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3586 \def\@Glsuserv@#1#2[#3] {%
```

```
3587   \@gls@field@link{#1}{#2}{\Glsentryuserv{#2}#3}%
```

```
3588 }
```

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

\GLSuserv

```
3589 \newrobustcmd*{\GLSuserv}{\@ifstar\@sGLSuserv\@GLSuserv}
```

Define the starred form:

```
3590 \newcommand*{\@sGLSuserv}[1] [] {\@GLSuserv[hyper=false,#1]}
```

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

```
3591 \newcommand*{\@GLSuserv}[2] [] {%
```

```
3592   \new@ifnextchar[{\@GLSuserv@{#1}{#2}}{\@GLSuserv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3593 \def\@GLSuserv@#1#2[#3] {%
```

```
3594   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryuserv{#2}#3}}%
```

```
3595 }
```

\glsuservi behaves like \gls except it always uses the value given by the userv6 key and it doesn't mark the entry as used.

`\glsuservi`

```
3596 \newrobustcmd*{\glsuservi}{\@ifstar\@sglsuservi\@glsuservi}
```

Define the starred form:

```
3597 \newcommand*{\@sglsuservi}[1] [] {\@glsuservi[hyper=false,#1]}
```

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

```
3598 \newcommand*{\@glsuservi}[2] [] {%
```

```
3599   \new@ifnextchar[{\@glsuservi@{#1}{#2}}{\@glsuservi@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3600 \def\@glsuservi@#1#2[#3] {%
```

```
3601   \@gls@field@link{#1}{#2}{\glsentryuservi{#2}#3}%
```

```
3602 }
```

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

`\Glsuservi`

```
3603 \newrobustcmd*{\Glsuservi}{\@ifstar\@sGlsuservi\@Glsuservi}
```

Define the starred form:

```
3604 \newcommand*{\@sGlsuservi}[1] [] {\@Glsuservi[hyper=false,#1]}
```

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

```
3605 \newcommand*{\@Glsuservi}[2] [] {%
```

```
3606   \new@ifnextchar[{\@Glsuservi@{#1}{#2}}{\@Glsuservi@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3607 \def\@Glsuservi@#1#2[#3] {%
```

```
3608   \@gls@field@link{#1}{#2}{\Glsentryuservi{#2}#3}%
```

```
3609 }
```

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

`\GLSuservi`

```
3610 \newrobustcmd*{\GLSuservi}{\@ifstar\@sGLSuservi\@GLSuservi}
```

Define the starred form:

```
3611 \newcommand*{\@sGLSuservi}[1] [] {\@GLSuservi[hyper=false,#1]}
```

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

```
3612 \newcommand*{\@GLSuservi}[2] [] {%
```

```
3613   \new@ifnextchar[{\@GLSuservi@{#1}{#2}}{\@GLSuservi@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3614 \def\@GLSuservi@#1#2[#3] {%
```

```
3615   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryuservi{#2}#3}}%
```

```
3616 }
```

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

`\acrshort`

```
3617 \newrobustcmd*{\acrshort}{\@ifstar\s@acrshort\ns@acrshort}
```

Define the starred form:

```
3618 \newcommand*{\s@acrshort}[2] [] {%
3619   \new@ifnextchar[{\@acrshort{hyper=false,#1}{#2}}{%
3620     {\@acrshort{hyper=false,#1}{#2} []}%
3621 }
```

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

```
3622 \newcommand*{\ns@acrshort}[2] [] {%
3623   \new@ifnextchar[{\@acrshort{#1}{#2}}{\@acrshort{#1}{#2} []}%
3624 }
```

Read in the final optional argument:

```
3625 \def\@acrshort#1#2[#3] {%
3626   \glsdoifexists{#2}%
3627   {%
3628     \edef\@glo@type{\glsentrytype{#2}}%

3629     \let\glsifplural\@secondoftwo
3630     \let\glscapscase\@firstofthree
3631     \let\glsinsert\@empty
3632     \def\glscustomtext{%
3633       \acronymfont{\glsentryshort{#2}}#3%
3634     }%
```

Call `\@gls@link`

```
3635   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3636   }%
3637 }
```

`\Acrshort`

```
3638 \newrobustcmd*{\Acrshort}{\@ifstar\s@Acrshort\ns@Acrshort}
```

Define the starred form:

```
3639 \newcommand*{\s@Acrshort}[2] [] {%
3640   \new@ifnextchar[{\@Acrshort{hyper=false,#1}{#2}}{%
3641     {\@Acrshort{hyper=false,#1}{#2} []}%
3642 }
```

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

```
3643 \newcommand*{\ns@Acrshort}[2] [] {%
3644   \new@ifnextchar[{\@Acrshort{#1}{#2}}{\@Acrshort{#1}{#2} []}%
3645 }
```

Read in the final optional argument:

```

3646 \def\@Acrshort#1#2[#3]{%
3647   \glsdoifexists{#2}%
3648   {%
3649     \edef\@glo@type{\glsentrytype{#2}}%

3650     \def\glslabel{#2}%
3651     \let\glsifplural\@secondoftwo
3652     \let\glscapscase\@secondofthree
3653     \let\glsinsert\@empty
3654     \def\glscustomtext{%
3655       \acronymfont{\glsentryshort{#2}}#3%
3656     }%

```

Call \@gls@link

```

3657   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3658 }%
3659 }

```

\ACRshort

```

3660 \newrobustcmd*{\ACRshort}{\@ifstar\s@ACRshort\ns@ACRshort}

```

Define the starred form:

```

3661 \newcommand*\s@ACRshort[2][{}]{%
3662   \new@ifnextchar[{\@ACRshort{hyper=false,#1}{#2}}%
3663   {\@ACRshort{hyper=false,#1}{#2}[]}%
3664 }

```

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

```

3665 \newcommand*\ns@ACRshort[2][{}]{%
3666   \new@ifnextchar[{\@ACRshort{#1}{#2}}{\@ACRshort{#1}{#2}[]}%
3667 }

```

Read in the final optional argument:

```

3668 \def\@ACRshort#1#2[#3]{%
3669   \glsdoifexists{#2}%
3670   {%
3671     \edef\@glo@type{\glsentrytype{#2}}%

3672     \def\glslabel{#2}%
3673     \let\glsifplural\@secondoftwo
3674     \let\glscapscase\@thirdofthree
3675     \let\glsinsert\@empty
3676     \def\glscustomtext{%
3677       \mfirstucMakeUppercase{\acronymfont{\glsentryshort{#2}}#3}%
3678     }%

```

Call \@gls@link

```

3679   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3680 }%
3681 }

```

Short plural:

`\acrshortpl`

```
3682 \newrobustcmd*{\acrshortpl}{\@ifstar\s@acrshortpl\ns@acrshortpl}
```

Define the starred form:

```
3683 \newcommand*{\s@acrshortpl}[2] [] {%
3684   \new@ifnextchar[{\@acrshortpl{hyper=false,#1}{#2}}{%
3685     {\@acrshortpl{hyper=false,#1}{#2} []}%
3686 }
```

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

```
3687 \newcommand*{\ns@acrshortpl}[2] [] {%
3688   \new@ifnextchar[{\@acrshortpl{#1}{#2}}{\@acrshortpl{#1}{#2} []}%
3689 }
```

Read in the final optional argument:

```
3690 \def\@acrshortpl#1#2[#3] {%
3691   \glsdoifexists{#2}%
3692   {%
3693     \edef\@glo@type{\glsentrytype{#2}}%
3694     \def\glslabel{#2}%
3695     \let\glsifplural\@firstoftwo
3696     \let\glscapscase\@firstofthree
3697     \let\glsinsert\@empty
3698     \def\glscustomtext{%
3699       \acronymfont{\glsentryshortpl{#2}}#3%
3700     }%
```

Call `\@gls@link`

```
3701   \@gls@link{#1}{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3702   }%
3703 }
```

`\Acrshortpl`

```
3704 \newrobustcmd*{\Acrshortpl}{\@ifstar\s@Acrshortpl\ns@Acrshortpl}
```

Define the starred form:

```
3705 \newcommand*{\s@Acrshortpl}[2] [] {%
3706   \new@ifnextchar[{\@Acrshortpl{hyper=false,#1}{#2}}{%
3707     {\@Acrshortpl{hyper=false,#1}{#2} []}%
3708 }
```

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

```
3709 \newcommand*{\ns@Acrshortpl}[2] [] {%
3710   \new@ifnextchar[{\@Acrshortpl{#1}{#2}}{\@Acrshortpl{#1}{#2} []}%
3711 }
```



Read in the final optional argument:

```
3712 \def\@Acrshortpl#1#2[#3]{%
3713   \glsdoifexists{#2}%
3714   {%
3715     \edef\@glo@type{\glsentrytype{#2}}%
3716     \def\glslabel{#2}%
3717     \let\glsifplural\@firstoftwo
3718     \let\glscapscase\@secondofthree
3719     \let\glsinsert\@empty
3720     \def\glscustomtext{%
3721       \acronymfont{\glsentryshortpl{#2}}#3%
3722     }%
```

Call \@gls@link

```
3723   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3724   }%
3725 }
```

\ACRshortpl

```
3726 \newrobustcmd*{\ACRshortpl}{\@ifstar\s@ACRshortpl\@ns@ACRshortpl}
```

Define the starred form:

```
3727 \newcommand*\s@ACRshortpl[2][{}]{%
3728   \new@ifnextchar[{\@ACRshortpl{hyper=false,#1}{#2}}%
3729   {\@ACRshortpl{hyper=false,#1}{#2}[]}%
3730 }
```

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

```
3731 \newcommand*\@ns@ACRshortpl[2][{}]{%
3732   \new@ifnextchar[{\@ACRshortpl{#1}{#2}}{\@ACRshortpl{#1}{#2}[]}%
3733 }
```

Read in the final optional argument:

```
3734 \def\@ACRshortpl#1#2[#3]{%
3735   \glsdoifexists{#2}%
3736   {%
3737     \edef\@glo@type{\glsentrytype{#2}}%
3738     \def\glslabel{#2}%
3739     \let\glsifplural\@firstoftwo
3740     \let\glscapscase\@thirdofthree
3741     \let\glsinsert\@empty
3742     \def\glscustomtext{%
3743       \mfirstucMakeUppercase{\acronymfont{\glsentryshortpl{#2}}#3}%
3744     }%
```

Call \@gls@link

```
3745   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3746   }%
3747 }
```

`\acrlong`

```
3748 \newrobustcmd*{\acrlong}{\@ifstar\s@acrlong\ns@acrlong}
```

Define the starred form:

```
3749 \newcommand*{\s@acrlong}[2] [] {%
3750   \new@ifnextchar[{\@acrlong{hyper=false,#1}{#2}}%
3751     {\@acrlong{hyper=false,#1}{#2} []}%
3752 }
```

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

```
3753 \newcommand*{\ns@acrlong}[2] [] {%
3754   \new@ifnextchar[{\@acrlong{#1}{#2}}{\@acrlong{#1}{#2} []}%
3755 }
```

Read in the final optional argument:

```
3756 \def\@acrlong#1#2[#3] {%
3757   \glsdoifexists{#2}%
3758   {%
3759     \edef\@glo@type{\glsentrytype{#2}}%
3760     \def\glslabel{#2}%
3761     \let\glsifplural\@secondoftwo
3762     \let\glscapscase\@firstofthree
3763     \let\glsinsert\@empty
```

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

```
3764   \def\glscustomtext{%
3765     \glsentrylong{#2}#3%
3766   }%
```

Call `\@gls@link`

```
3767   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3768   }%
3769 }
```

`\Acrlong`

```
3770 \newrobustcmd*{\Acrlong}{\@ifstar\s@Acrlong\ns@Acrlong}
```

Define the starred form:

```
3771 \newcommand*{\s@Acrlong}[2] [] {%
3772   \new@ifnextchar[{\@Acrlong{hyper=false,#1}{#2}}%
3773     {\@Acrlong{hyper=false,#1}{#2} []}%
3774 }
```

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

```
3775 \newcommand*{\ns@Acrlong}[2] [] {%
3776   \new@ifnextchar[{\@Acrlong{#1}{#2}}{\@Acrlong{#1}{#2} []}%
3777 }
```

Read in the final optional argument:

```

3778 \def\@Acrlong#1#2[#3]{%
3779   \glsdoifexists{#2}%
3780   {%
3781     \edef\@glo@type{\glsentrytype{#2}}%

3782   \def\glslabel{#2}%
3783   \let\glsifplural\@secondoftwo
3784   \let\glscapscase\@secondofthree
3785   \let\glsinsert\@empty

```

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

```

3786   \def\glscustomtext{%
3787     \Glsentrylong{#2}#3%
3788   }%

```

Call \@gls@link

```

3789   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3790   }%
3791 }

```

\ACRlong

```

3792 \newrobustcmd*{\ACRlong}{\@ifstar\s@ACRlong\@ns@ACRlong}

```

Define the starred form:

```

3793 \newcommand*{\s@ACRlong}[2][]{%
3794   \new@ifnextchar[{\@ACRlong{hyper=false,#1}{#2}}{%
3795     {\@ACRlong{hyper=false,#1}{#2} []}%
3796 }

```

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

```

3797 \newcommand*{\ns@ACRlong}[2][]{%
3798   \new@ifnextchar[{\@ACRlong{#1}{#2}}{\@ACRlong{#1}{#2} []}%
3799 }

```

Read in the final optional argument:

```

3800 \def\@ACRlong#1#2[#3]{%
3801   \glsdoifexists{#2}%
3802   {%
3803     \edef\@glo@type{\glsentrytype{#2}}%

3804   \def\glslabel{#2}%
3805   \let\glsifplural\@secondoftwo
3806   \let\glscapscase\@thirdofthree
3807   \let\glsinsert\@empty

```

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

```

3808 \def\glscustomtext{%
3809 \mfirstucMakeUppercase{\glsentrylong{#2}#3}%
3810 }%
    Call \@gls@link
3811 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3812 }%
3813 }

```

Short plural:

\acrlongpl

```

3814 \newrobustcmd*{\acrlongpl}{\@ifstar\s@acrlongpl\ns@acrlongpl}

```

Define the starred form:

```

3815 \newcommand*{\s@acrlongpl}[2] [] {%
3816 \new@ifnextchar[{\@acrlongpl{hyper=false,#1}{#2}}%
3817 {\@acrlongpl{hyper=false,#1}{#2} []}%
3818 }

```

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

```

3819 \newcommand*{\ns@acrlongpl}[2] [] {%
3820 \new@ifnextchar[{\@acrlongpl{#1}{#2}}{\@acrlongpl{#1}{#2} []}%
3821 }

```

Read in the final optional argument:

```

3822 \def\@acrlongpl#1#2[#3] {%
3823 \glsoifexists{#2}%
3824 {%
3825 \edef\@glo@type{\glsentrytype{#2}}%
3826 \def\glslabel{#2}%
3827 \let\glsifplural\@firstoftwo
3828 \let\glscapscase\@firstofthree
3829 \let\glsinsert\@empty

```

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

```

3830 \def\glscustomtext{%
3831 \glsentrylongpl{#2}#3%
3832 }%

```

Call \@gls@link

```

3833 \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3834 }%
3835 }

```

\Acrlongpl

```

3836 \newrobustcmd*{\Acrlongpl}{\@ifstar\s@Acrlongpl\ns@Acrlongpl}

```

Define the starred form:

```
3837 \newcommand*{\s@Acrlongpl}[2] [] {%
3838   \new@ifnextchar[{\@Acrlongpl{hyper=false,#1}{#2}}{%
3839     {\@Acrlongpl{hyper=false,#1}{#2} []}%
3840 }
```

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

```
3841 \newcommand*{\ns@Acrlongpl}[2] [] {%
3842   \new@ifnextchar[{\@Acrlongpl{#1}{#2}}{\@Acrlongpl{#1}{#2} []}%
3843 }
```

Read in the final optional argument:

```
3844 \def\@Acrlongpl#1#2[#3] {%
3845   \glsdoifexists{#2}%
3846   {%
3847     \edef\@glo@type{\glsentrytype{#2}}%

3848     \def\glslabel{#2}%
3849     \let\glsifplural\@firstoftwo
3850     \let\glscapscase\@secondofthree
3851     \let\glsinsert\@empty
```

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

```
3852   \def\glscustomtext{%
3853     \Glsentrylongpl{#2}#3%
3854   }%
```

Call \@gls@link

```
3855   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3856   }%
3857 }
```

\ACRlongpl

```
3858 \newrobustcmd*{\ACRlongpl}{\@ifstar\s@ACRlongpl\ns@ACRlongpl}
```

Define the starred form:

```
3859 \newcommand*{\s@ACRlongpl}[2] [] {%
3860   \new@ifnextchar[{\@ACRlongpl{hyper=false,#1}{#2}}{%
3861     {\@ACRlongpl{hyper=false,#1}{#2} []}%
3862 }
```

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

```
3863 \newcommand*{\ns@ACRlongpl}[2] [] {%
3864   \new@ifnextchar[{\@ACRlongpl{#1}{#2}}{\@ACRlongpl{#1}{#2} []}%
3865 }
```

Read in the final optional argument:

```

3866 \def\@ACRlongpl#1#2[#3]{%
3867   \glsdoifexists{#2}%
3868   {%
3869     \edef\@glo@type{\glentrytype{#2}}%

3870   \def\glslabel{#2}%
3871   \let\glsifplural\@firstoftwo
3872   \let\glscapscase\@thirdofthree
3873   \let\glsinsert\@empty

```

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

```

3874   \def\glscustomtext{%
3875     \mfirstucMakeUppercase{\glentrylongpl{#2}#3}%
3876   }%

```

Call `\@gls@link`

```

3877   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
3878   }%
3879 }

```

### 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>}
```

```

3880 \newcommand*{\@gls@entry@field}[2]{%
3881   \csname glo@\glsdetoklabel{#1}@#2\endcsname
3882 }

```

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

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

```

3883 \newcommand*{\@Gls@entry@field}[2]{%
3884   \letcs\@glo@text{glo@\glsdetoklabel{#1}@#2}%
3885   \xmakefirstuc{\@glo@text}%
3886 }

```

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
3887 \newcommand*{\glsentryname}[1]{\@gls@entry@field{#1}{name}}
```

```
\Glsentryname
3888 \newrobustcmd*{\Glsentryname}[1]{%
3889   \@Gls@entry@field{#1}{name}%
3890 }
```

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
3891 \newcommand*{\glsentrydesc}[1]{\@gls@entry@field{#1}{desc}}
```

```
\Glsentrydesc
3892 \newrobustcmd*{\Glsentrydesc}[1]{%
3893   \@Gls@entry@field{#1}{desc}%
3894 }
```

Plural form:

```
\glsentrydescplural
3895 \newcommand*{\glsentrydescplural}[1]{%
3896   \@gls@entry@field{#1}{descplural}%
3897 }
```

```
\Glsentrydescplural
3898 \newrobustcmd*{\Glsentrydescplural}[1]{%
3899   \@Gls@entry@field{#1}{descplural}%
3900 }
```

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
3901 \newcommand*{\glsentrytext}[1]{\@gls@entry@field{#1}{text}}
```

```
\Glsentrytext
3902 \newrobustcmd*{\Glsentrytext}[1]{%
3903   \@Gls@entry@field{#1}{text}%
3904 }
```

Get the plural form:

```
\glsentryplural
3905 \newcommand*{\glsentryplural}[1]{%
3906   \@gls@entry@field{#1}{plural}%
3907 }
```

`\Glsentryplural`

```
3908 \newrobustcmd*{\Glsentryplural}[1]{%
3909   \@Gls@entry@field{#1}{plural}%
3910 }
```

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

`\glsentrysymbol`

```
3911 \newcommand*{\glsentrysymbol}[1]{%
3912   \@Gls@entry@field{#1}{symbol}%
3913 }
```

`\Glsentrysymbol`

```
3914 \newrobustcmd*{\Glsentrysymbol}[1]{%
3915   \@Gls@entry@field{#1}{symbol}%
3916 }
```

Plural form:

`\glsentrysymbolplural`

```
3917 \newcommand*{\glsentrysymbolplural}[1]{%
3918   \@Gls@entry@field{#1}{symbolplural}%
3919 }
```

`\Glsentrysymbolplural`

```
3920 \newrobustcmd*{\Glsentrysymbolplural}[1]{%
3921   \@Gls@entry@field{#1}{symbolplural}%
3922 }
```

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`

```
3923 \newcommand*{\glsentryfirst}[1]{%
3924   \@Gls@entry@field{#1}{first}%
3925 }
```

`\Glsentryfirst`

```
3926 \newrobustcmd*{\Glsentryfirst}[1]{%
3927   \@Gls@entry@field{#1}{first}%
3928 }
```

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

`\glsentryfirstplural`

```
3929 \newcommand*{\glsentryfirstplural}[1]{%
3930   \@Gls@entry@field{#1}{firstpl}%
3931 }
```



Glsentryfirstplural

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

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

\glsentrytype

```
3935 \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

```
3936 \newcommand*{\glsentrysort}[1]{%
3937   \@Gls@entry@field{#1}{sort}%
3938 }
```

\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.

```
3939 \newcommand*{\glsentryuseri}[1]{%
3940   \@Gls@entry@field{#1}{useri}%
3941 }
```

\Glsentryuseri

```
3942 \newrobustcmd*{\Glsentryuseri}[1]{%
3943   \@Gls@entry@field{#1}{useri}%
3944 }
```

\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.

```
3945 \newcommand*{\glsentryuserii}[1]{%
3946   \@Gls@entry@field{#1}{userii}%
3947 }
```

\Glsentryuserii

```
3948 \newrobustcmd*{\Glsentryuserii}[1]{%
3949   \@Gls@entry@field{#1}{userii}%
3950 }
```

\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.

```
3951 \newcommand*{\glsentryuseriii}[1]{%
3952   \@Gls@entry@field{#1}{useriii}%
3953 }
```

\Glsentryuseriii

```
3954 \newrobustcmd*{\Glsentryuseriii}[1]{%
3955   \@Gls@entry@field{#1}{useriii}%
3956 }
```

\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.

```
3957 \newcommand*{\glsentryuseriv}[1]{%
3958   \@Gls@entry@field{#1}{useriv}%
3959 }
```

\Glsentryuseriv

```
3960 \newrobustcmd*{\Glsentryuseriv}[1]{%
3961   \@Gls@entry@field{#1}{useriv}%
3962 }
```

\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.

```
3963 \newcommand*{\glsentryuserv}[1]{%
3964   \@Gls@entry@field{#1}{userv}%
3965 }
```

\Glsentryuserv

```
3966 \newrobustcmd*{\Glsentryuserv}[1]{%
3967   \@Gls@entry@field{#1}{userv}%
3968 }
```

\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.

```
3969 \newcommand*{\glsentryuservi}[1]{%
3970   \@Gls@entry@field{#1}{uservi}%
3971 }
```

\Glsentryuservi

```
3972 \newrobustcmd*{\Glsentryuservi}[1]{%
3973   \@Gls@entry@field{#1}{uservi}%
3974 }
```

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

```
3975 \newcommand*{\glsentryshort}[1]{\@Gls@entry@field{#1}{short}}
```

\Glsentryshort

```
3976 \newrobustcmd*{\Glsentryshort}[1]{%
3977   \@Gls@entry@field{#1}{short}%
3978 }
```

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

```

3979 \newcommand*{\glsentryshortpl}[1]{\@gls@entry@field{#1}{shortpl}}

\Glsentryshortpl
3980 \newrobustcmd*{\Glsentryshortpl}[1]{%
3981   \@Gls@entry@field{#1}{shortpl}}%
3982 }

\glsentrylong Get the long key (as specified by the long the entry was defined). The argument
is the label associated with the entry.
3983 \newcommand*{\glsentrylong}[1]{\@gls@entry@field{#1}{long}}

\Glsentrylong
3984 \newrobustcmd*{\Glsentrylong}[1]{%
3985   \@Gls@entry@field{#1}{long}}%
3986 }

\glsentrylongpl Get the long plural key (as specified by the longplural the entry was defined).
The argument is the label associated with the entry.
3987 \newcommand*{\glsentrylongpl}[1]{\@gls@entry@field{#1}{longpl}}

\Glsentrylongpl
3988 \newrobustcmd*{\Glsentrylongpl}[1]{%
3989   \@Gls@entry@field{#1}{longpl}}%
3990 }

Short cut macros to access full form:

\glsentryfull
3991 \newcommand*{\glsentryfull}[1]{%
3992   \acrfullformat{\glsentrylong{#1}}{\acronymfont{\glsentryshort{#1}}}%
3993 }

\Glsentryfull
3994 \newrobustcmd*{\Glsentryfull}[1]{%
3995   \acrfullformat{\Glsentrylong{#1}}{\acronymfont{\glsentryshort{#1}}}%
3996 }

\glsentryfullpl
3997 \newcommand*{\glsentryfullpl}[1]{%
3998   \acrfullformat{\glsentrylongpl{#1}}{\acronymfont{\glsentryshortpl{#1}}}%
3999 }

\Glsentryfullpl
4000 \newrobustcmd*{\Glsentryfullpl}[1]{%
4001   \acrfullformat{\Glsentrylongpl{#1}}{\acronymfont{\glsentryshortpl{#1}}}%
4002 }

```

`\glsentrynumberlist` Displays the number list as is.

```

4003 \newcommand*{\glsentrynumberlist}[1]{%
4004   \glsdoifexists{#1}%
4005   {%
4006     \@gls@entry@field{#1}{numberlist}%
4007   }%
4008 }
```

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

```

4009 \@ifpackageloaded{hyperref} {%
4010   \newcommand*{\glsdisplaynumberlist}[1]{%
4011     \GlossariesWarning
4012     {%
4013       \string\glsdisplaynumberlist\space
4014       doesn't work with hyperref.^^JUsing
4015       \string\glsentrynumberlist\space instead%
4016     }%
4017     \glsentrynumberlist{#1}%
4018   }%
4019 }%
4020 {%
4021   \newcommand*{\glsdisplaynumberlist}[1]{%
4022     \glsdoifexists{#1}%
4023     {%
4024       \bgroup

4025       \edef\@glo@label{\glsdetoklabel{#1}}%
4026       \let\@org@glsglnumberformat\glsglnumberformat
4027       \def\glsglnumberformat##1{##1}%
4028       \protected@edef\the@numberlist{%
4029         \csname glo@\@glo@label @numberlist\endcsname}%
4030       \def\@glsgl@numlist@sep{}%
4031       \def\@glsgl@numlist@nextsep{}%
4032       \def\@glsgl@numlist@lastsep{}%
4033       \def\@glsgl@thislist{}%
4034       \def\@glsgl@donext@def{}%
4035       \renewcommand\do[1]{%
4036         \protected@edef\@glsgl@thislist{%
4037           \@glsgl@thislist
4038           \noexpand\@glsgl@numlist@sep
4039           ##1%
4040         }%
4041         \let\@glsgl@numlist@sep\@glsgl@numlist@nextsep
4042         \def\@glsgl@numlist@nextsep{\glsgl@numlist@sep}%
4043         \@glsgl@donext@def
4044         \def\@glsgl@donext@def{%
4045           \def\@glsgl@numlist@lastsep{\glsgl@numlist@lastsep}%
4046         }%
4047       }%
4048     }%
4049   }%
4050 }
```

```

4048      \expandafter \glsnumlistparser \expandafter{\the@numberlist}%
4049      \let\@gls@numlist@sep\@gls@numlist@lastsep
4050      \@gls@thislist
4051      \egroup
4052    }%
4053  }
4054 }

```

`\glsnumlistsep`

```
4055 \newcommand*{\glsnumlistsep}{, }
```

`\glsnumlistlastsep`

```
4056 \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.

```

4057 \newcommand*{\glshyperlink}[2][\glsentrytext{\@glo@label}]{%
4058   \def\@glo@label{#2}%
4059   \@glslink{\glo@linkprefix\glsdetoklabel{#2}}{#1}}

```

## 1.11 Adding an entry to the glossary without generating text

The following keys are provided for `\glsadd` and `\glsaddall`:

```
4060 \define@key{glossadd}{counter}{\def\@gls@counter{#1}}
```

```
4061 \define@key{glossadd}{format}{\def\@glsnumberformat{#1}}
```

This key is only used by `\glsaddall`:

```
4062 \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`

```

4063 \newrobustcmd*{\glsadd}[2][ ]{%
4064   \glsdoifexists{#2}%
4065   {%
4066     \def\@glsnumberformat{glsnumberformat}%
4067     \edef\@gls@counter{\csname glo@glsdetoklabel{#2}@counter\endcsname}%
4068     \setkeys{glossadd}{#1}%

```

Store the entry's counter in `\theglentrycounter`

```
4069 \gls@saveentrycounter
4070 \do@wrglossary{#2}%
4071 }%
4072 }
```

`\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`

```
4073 \newrobustcmd*{\glsaddall}[1] [] {%
4074   \edef\@glo@type{\@glo@types}%
4075   \setkeys{glossadd}{#1}%
4076   \forallglsentries[\@glo@type]{\@glo@entry}{%
4077     \glsadd[#1]{\@glo@entry}%
4078   }%
4079 }
```

`\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.

```
4080 \newrobustcmd*{\glsaddallunused}[1] [\@glo@types] {%
4081   \forallglsentries[#1]{\@glo@entry}%
4082   {%
4083     \ifglsused{\@glo@entry}{\@glsadd[format=@gobble]{\@glo@entry}}%
4084   }%
4085 }
```

## 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.

```
4086 \edef\glsopenbrace{\expandafter\@gobble\string\{}
```

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

```
4087 \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.

```
4088 \edef\glsquote#1{\string"#1\string"}
```

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

```
4089 \ifglsxindy
4090   \newcommand*{\@glsfirstletter}{A}
4091 \fi
```

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

```
4092 \ifglsxindy
4093   \newcommand*{\GlsSetXdyFirstLetterAfterDigits}[1]{%
4094     \renewcommand*{\@glsfirstletter}{#1}}
4095 \else
4096   \newcommand*{\GlsSetXdyFirstLetterAfterDigits}[1]{%
4097     \glsnoxywarning\GlsSetXdyFirstLetterAfterDigits}
4098 \fi
```

\@glsminrange Define the minimum number of successive location references to merge into a range.

```
4099 \newcommand*{\@glsminrange}{2}
```

etXdyMinRangeLength Set the minimum range length. The value must either be none or a positive integer. The glossaries package doesn't check if the argument is valid, that is left to xindy.

```
4100 \ifglsxindy
4101   \newcommand*{\GlsSetXdyMinRangeLength}[1]{%
4102     \renewcommand*{\@glsminrange}{#1}}
4103 \else
4104   \newcommand*{\GlsSetXdyMinRangeLength}[1]{%
4105     \glsnoxywarning\GlsSetXdyMinRangeLength}
4106 \fi
```

\writeist

```
4107 \ifglsxindy
```

Code to use if xindy is required.

```

4108 \def\writeist{%
    Define write register if not already defined
4109 \ifundef{\glswrite}{\newwrite\glswrite}{}%
    Update attributes list
4110 \@gls@addpredefinedattributes
    Open the file.
4111 \openout\glswrite=\istfilename
    Write header comment at the start of the file
4112 \write\glswrite{;; xindy style file created by the glossaries
4113 package}%
4114 \write\glswrite{;; for document '\jobname' on
4115 \the\year-\the\month-\the\day}%
    Specify the required styles
4116 \write\glswrite{^^J; required styles^^J}
4117 \@for\@xdystyle:=\@xdyrequiredstyles\do{%
4118 \ifx\@xdystyle\@empty
4119 \else
4120 \protected@write\glswrite{{(require
4121 \string"\@xdystyle.xdy\string")}}%
4122 \fi
4123 }%
    List the allowed attributes (possible values used by the format key)
4124 \write\glswrite{^^J%
4125 ; list of allowed attributes (number formats)^^J}%
4126 \write\glswrite{(define-attributes ((\@xdyattributes)))}%
    Define any additional alphabets
4127 \write\glswrite{^^J; user defined alphabets^^J}%
4128 \write\glswrite{\@xdyuseralphabets}%
    Define location classes.
4129 \write\glswrite{^^J; location class definitions^^J}%
    As from version 3.0, locations are now specified as {\langle Hprefix\rangle}{\langle number\rangle}, so
    need to add all possible combinations of location types.
4130 \@for\@gls@classI:=\@gls@xdy@locationlist\do{%
    Case were \langle Hprefix\rangle is empty:
4131 \protected@write\glswrite{{(define-location-class
4132 \string"\@gls@classI\string"^^J\space\space\space
4133 (
4134 :sep "{}{"
4135 \csname @gls@xdy@Lclass@\@gls@classI\endcsname\space
4136 :sep "}"
4137 )
4138 ^^J\space\space\space

```



```

4139         :min-range-length \@glsminrange^^J%
4140     )
4141 }%

```

Nested iteration over all classes:

```

4142 {%
4143     \@for\@gls@classII:=\@gls@xdy@locationlist\do{%
4144         \protected@write\glswrite{}\{(define-location-class
4145             \string\@gls@classII-\@gls@classI\string"
4146             ^^J\space\space\space
4147             (
4148                 :sep "{"
4149                 \csname @gls@xdy@Lclass@\@gls@classII\endcsname\space
4150                 :sep "{{"
4151                 \csname @gls@xdy@Lclass@\@gls@classI\endcsname\space
4152                 :sep "}"
4153             )
4154             ^^J\space\space\space
4155             :min-range-length \@glsminrange^^J%
4156         )
4157     }%
4158 }%
4159 }%
4160 }%

```

User defined location classes (needs checking for new location format).

```

4161 \write\glswrite{^^J; user defined location classes}%
4162 \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.)

```

4163 \write\glswrite{^^J; define cross-reference class^^J}%
4164 \write\glswrite{(define-crossref-class \string"see\string"
4165     :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.)

```

4166 \write\glswrite{(markup-crossref-list
4167     :class \string"see\string"^^J\space\space\space
4168     :open \string"\string\glsseeformat\string"
4169     :close \string"{}\string")}%

```

List the order to sort the classes.

```

4170 \write\glswrite{^^J; define the order of the location classes}%
4171 \write\glswrite{(define-location-class-order
4172     (\@xdylocationclassorder))}%

```

Specify what to write to the start and end of the glossary file.

```

4173 \write\glswrite{^^J; define the glossary markup^^J}%

```

```

4174 \write\glswrite{(markup-index^^J\space\space\space
4175 :open \string"\string
4176 \glossarysection[\string\glossarytoctitle]{\string
4177 \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)

```

4178 \@for\@this@ctr:=\@xdycounters\do{%
4179 {%
4180 \for\@this@attr:=\@xdyattributelist\do{%
4181 \protected\write\glswrite{ }\string\providecommand*%
4182 \expandafter\string
4183 \csname glsX\@this@ctr X\@this@attr\endcsname[2]%
4184 {%
4185 \string\setentrycounter
4186 [\expandafter\@gobble\string\#1]{\@this@ctr}%
4187 \expandafter\string
4188 \csname\@this@attr\endcsname
4189 {\expandafter\@gobble\string\#2}%
4190 }%
4191 }%
4192 }%
4193 }%
4194 }%

```

Add the end part of the open tag and the rest of the markup-index information:

```

4195 \write\glswrite{%
4196 \string\begin
4197 {theglossary}\string\glossaryheader\string~n\string" ^^J\space
4198 \space\space:close \string"\expandafter\@gobble
4199 \string%\string~n\string
4200 \end{theglossary}\string\glossarypostamble
4201 \string~n\string" ^^J\space\space\space
4202 :tree)}}%

```

Specify what to put between letter groups

```

4203 \write\glswrite{(markup-letter-group-list
4204 :sep \string"\string\glsgrpskip\string~n\string"))%

```

Specify what to put between entries

```

4205 \write\glswrite{(markup-indexentry
4206 :open \string"\string\relax \string\glsresetentrylist
4207 \string~n\string"))%

```

Specify how to format entries

```

4208 \write\glswrite{(markup-locclass-list :open
4209 \string"\glsopenbrace\string\glossaryentrynumbers
4210 \glsopenbrace\string\relax\space \string"^^J\space\space\space
4211 :sep \string", \string"
4212 :close \string"\glsclosebrace\glsresetentrylist\string"))%

```

Specify how to separate location numbers

```
4213 \write\glswrite{(markup-locref-list
4214 :sep \string\string\delimN\space\string)}}%
```

Specify how to indicate location ranges

```
4215 \write\glswrite{(markup-range
4216 :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.

```
4217 \@onelevel@sanitize\gls@suffiF
4218 \@onelevel@sanitize\gls@suffiFF

4219 \ifx\gls@suffiF\@empty
4220 \else
4221 \write\glswrite{(markup-range
4222 :close "\gls@suffiF" :length 1 :ignore-end)}}%
4223 \fi
4224 \ifx\gls@suffiFF\@empty
4225 \else
4226 \write\glswrite{(markup-range
4227 :close "\gls@suffiFF" :length 2 :ignore-end)}}%
4228 \fi
```

Specify how to format locations.

```
4229 \write\glswrite{^^J; define format to use for locations^^J}%
4230 \write\glswrite{\@xdylocref}}%
```

Specify how to separate letter groups.

```
4231 \write\glswrite{^^J; define letter group list format^^J}%
4232 \write\glswrite{(markup-letter-group-list
4233 :sep \string\string\glsgroupskip\string~n\string)}}%
```

Define letter group headings.

```
4234 \write\glswrite{^^J; letter group headings^^J}%
4235 \write\glswrite{(markup-letter-group
4236 :open-head \string\string\glsgroupheading
4237 \glsopenbrace\string^^J\space\space\space
4238 :close-head \string\glsclosebrace\string)}}%
```

Define additional letter groups.

```
4239 \write\glswrite{^^J; additional letter groups^^J}%
4240 \write\glswrite{\@xdylettergroups}}%
```

Define additional sort rules

```
4241 \write\glswrite{^^J; additional sort rules^^J}
4242 \write\glswrite{\@xdysortrules}}%
```

Close the style file

```
4243 \closeout\glswrite
```

Suppress any further calls.

```
4244 \let\writeist\relax
4245 }
4246 \else
```

Code to use if makeindex is required.

```
4247 \edef\@gls@actualchar{\string?}
4248 \edef\@gls@encapchar{\string|}
4249 \edef\@gls@levelchar{\string!}
4250 \edef\@gls@quotechar{\string"}
4251 \def\writeist{\relax
4252 \ifundef{glswrite}{\newwrite\glswrite}{\relax
4253 \openout\glswrite=\istfilename
4254 \write\glswrite{\expandafter\@gobble\string\% makeindex style file
4255 created by the glossaries package}
4256 \write\glswrite{\expandafter\@gobble\string\% for document
4257 '\jobname' on \the\year-\the\month-\the\day}
4258 \write\glswrite{actual '\@gls@actualchar'}
4259 \write\glswrite{encap '\@gls@encapchar'}
4260 \write\glswrite{level '\@gls@levelchar'}
4261 \write\glswrite{quote '\@gls@quotechar'}
4262 \write\glswrite{keyword \string"\string\glossaryentry\string"}
4263 \write\glswrite{preamble \string"\string\glossarysection[\string
4264 \glossarytoctitle]{\string\glossarytitle}\string
4265 \glossarypreamble\string\n\string\begin{theglossary}\string
4266 \glossaryheader\string\n\string"}
4267 \write\glswrite{postamble \string"\string%\string\n\string
4268 \end{theglossary}\string\glossarypostamble\string\n
4269 \string"}
4270 \write\glswrite{group_skip \string"\string\glsgroupskip\string\n
4271 \string"}
4272 \write\glswrite{item_0 \string"\string%\string\n\string"}
4273 \write\glswrite{item_1 \string"\string%\string\n\string"}
4274 \write\glswrite{item_2 \string"\string%\string\n\string"}
4275 \write\glswrite{item_01 \string"\string%\string\n\string"}
4276 \write\glswrite{item_x1
4277 \string"\string\relax \string\glsresetentrylist\string\n
4278 \string"}
4279 \write\glswrite{item_12 \string"\string%\string\n\string"}
4280 \write\glswrite{item_x2
4281 \string"\string\relax \string\glsresetentrylist\string\n
4282 \string"}

4283 \write\glswrite{delim_0 \string"\string\{\string
4284 \glossaryentrynumbers\string\{\string\relax \string"}
4285 \write\glswrite{delim_1 \string"\string\{\string
4286 \glossaryentrynumbers\string\{\string\relax \string"}
4287 \write\glswrite{delim_2 \string"\string\{\string
4288 \glossaryentrynumbers\string\{\string\relax \string"}
4289 \write\glswrite{delim_t \string"\string\}\string\}\string"}
```

```

4290 \write\glswrite{delim_n \string"\string\\delimN \string"}
4291 \write\glswrite{delim_r \string"\string\\delimR \string"}
4292 \write\glswrite{headings_flag 1}
4293 \write\glswrite{heading_prefix
4294   \string"\string\glsgroupheading\string\{\string"}
4295 \write\glswrite{heading_suffix
4296   \string"\string\\}\string\\relax
4297   \string\\glsgroupresetentrylist \string"}
4298 \write\glswrite{symhead_positive \string"glssymbols\string"}
4299 \write\glswrite{numhead_positive \string"glsglnumbers\string"}
4300 \write\glswrite{page_compositor \string"\glsglcompositor\string"}
4301 \@glsgl@escbsdq\glsgl@suffixF
4302 \@glsgl@escbsdq\glsgl@suffixFF
4303 \ifx\glsgl@suffixF\@empty
4304 \else
4305   \write\glswrite{suffix_2p \string"\glsgl@suffixF\string"}
4306 \fi
4307 \ifx\glsgl@suffixFF\@empty
4308 \else
4309   \write\glswrite{suffix_3p \string"\glsgl@suffixFF\string"}
4310 \fi
4311 \closeout\glswrite
4312 \let\writeist\relax
4313 }
4314 \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`

```

4315 \newcommand{\noist}{%
  Update attributes list
4316   \@glsgl@addpredefinedattributes
4317   \let\writeist\relax
4318 }

```

`\@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  $\text{\TeX}$  is trying to write to a non-existent file). The relevant glossary must be defined prior to using `\@makeglossary`.

`\@makeglossary`

```

4319 \newcommand*{\@makeglossary}[1]{%
4320   \ifglossaryexists{#1}%
4321   {%
      Only create a new write if savewrites=false otherwise create a token to collect
      the information.
4322     \ifglssavewrites
4323       \expandafter\newtoks\csname glo@#1@filetok\endcsname
4324     \else
4325       \expandafter\newwrite\csname glo@#1@file\endcsname
4326       \expandafter\@glsopenfile\csname glo@#1@file\endcsname{#1}%
4327     \fi
4328     \@gls@renewglossary
4329     \writeist
4330   }%
4331   {%
4332     \PackageError{glossaries}%
4333     {Glossary type ‘#1’ not defined}%
4334     {New glossaries must be defined before using \string\makeglossary}%
4335   }%
4336 }

```

`\@glsopenfile` Open write file associated with the given glossary.

```

4337 \newcommand*{\@glsopenfile}[2]{%
4338   \immediate\openout#1=\jobname.\csname @glotype@#2@out\endcsname
4339   \PackageInfo{glossaries}{Writing glossary file
4340     \jobname.\csname @glotype@#2@out\endcsname}%
4341 }

```

`\@nomakeglossaries` Issue warning that `\makeglossaries` hasn't been used.

```

4342 \newcommand*{\@warn@nomakeglossaries}{}

```

Only use this if warning if `\printglossary` has been used without `\makeglossaries`

```

4343 \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`

```

4344 \newcommand*{\makeglossaries}{%

```

Define the write used for style file also used for all other output files if `savewrites=true`.

```

4345   \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.

```

4346 \protected@write\@auxout{}\string\providecommand\string\@glsorder[1]{}
4347 \protected@write\@auxout{}\string\providecommand\string\@istfilename[1]{}
4348 % Write the name of the style file to the aux file
4349 % (needed by \app{makeglossaries})
4350 % \begin{macrocode}
4351 \protected@write\@auxout{}\string\@istfilename{\istfilename}}%
4352 \protected@write\@auxout{}\string\@glsorder{\glsorder}}

Iterate through each glossary type and activate it.

4353 \@for\@glo@type:=\@glo@types\do{%
4354 \ifthenelse{\equal{\@glo@type}{}}{}{%
4355 \@makeglossary{\@glo@type}}%
4356 }%

New glossaries must be created before \makeglossaries so disable \newglossary.

4357 \renewcommand*\newglossary[4][]{%
4358 \PackageError{glossaries}{New glossaries
4359 must be created before \string\makeglossaries}{You need
4360 to move \string\makeglossaries\space after all your
4361 \string\newglossary\space commands}}%

Any subsequent instances of this command should have no effect

4362 \let\@makeglossary\relax
4363 \let\makeglossary\relax
4364 \let\makeglossaries\relax

Disable all commands that have no effect after \makeglossaries

4365 \@disable@onlypremakeg

Allow see key:

4366 \let\gls@checkseeallowed\relax

Suppress warning about no \makeglossaries

4367 \let\warn@nomakeglossaries\relax

Activate warning about missing \printglossary

4368 \def\warn@noprintglossary{%
4369 \GlossariesWarningNoLine{No \string\printglossary\space
4370 or \string\printglossaries\space
4371 found.^^J(Remove \string\makeglossaries\space if you don't want
4372 any glossaries.)^^JThis document will not have a glossary}%
4373 }%

Declare list parser for \glsdisplaynumberlist

4374 \ifglssavenumberlist
4375 \edef\@gls@dodolistparser{\noexpand\DeclareListParser
4376 {\noexpand\glsnumlistparser}{\delimN}}%
4377 \@gls@dodolistparser
4378 \fi

Prevent user from also using \makenoidxglossaries

4379 \let\makenoidxglossaries\@no@makeglossaries

```

Prohibit sort key in printgloss family:

```
4380 \renewcommand*{\@printgloss@setsort}{%
4381   \let\@glo@assign@sortkey\@glo@no@assign@sortkey
4382 }%
4383 }
```

Must occur in the preamble:

```
4384 \@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`

```
4385 \let\makeglossary\makeglossaries
```

If `\makeglossaries` hasn't been used, issue a warning. Also issue a warning if neither `\printglossaries` nor `\printglossary` have been used.

```
4386 \AtEndDocument{%
4387   \warn@nomakeglossaries
4388   \warn@noprintglossary
4389 }
```

`\makenoidxglossaries` Analogous to `\makeglossaries` this activates the commands needed for `\printnoidxglossary`

```
4390 \newcommand*{\makenoidxglossaries}{%
```

Redefine empty glossary warning:

```
4391   \renewcommand{\@gls@noref@warn}[1]{%
4392     \GlossariesWarning{Empty glossary for
4393       \string\printnoidxglossary[type={##1}].
4394     Rerun may be required (or you may have forgotten to use
4395     commands like \string\gls).}%
4396   }%
```

Don't escape makeindex/xindy characters

```
4397   \let\@gls@checkmkidxchars\@gobble
```

Write glossary information to aux instead of glossary files

```
4398   \let\@@do@wrglossary\gls@noidxglossary
```

Switch on group headings that use the character code:

```
4399   \let\@gls@getgrouptitle\@gls@nidx@getgrouptitle
```

Allow see key:

```
4400   \let\gls@checkseeallowed\relax
```



Redefine cross-referencing macro:

```

4401 \renewcommand{\@do@seeglossary}[2]{%
4402   \edef\@gls@label{\glsdetoklabel{##1}}%
4403   \protected@write\@auxout{}\@auxout{}{%
4404     \string\@gls@reference
4405     {\csname glo@\@gls@label @type\endcsname}%
4406     {\@gls@label}%
4407     {%
4408       \string\glsseeformat##2}%
4409     }%
4410   }%
4411 }%

```

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.

```

4412 \AtBeginDocument
4413 {%
4414   \write\@auxout{\string\providecommand\string\@gls@reference[3]{}}%
4415 }%

```

Change warning about no glossaries

```

4416 \def\warn@noprintglossary{%
4417   \GlossariesWarningNoLine{No \string\printnoidxglossary\space
4418     or \string\printnoidxglossaries ^^J
4419     found. (Remove \string\makenoidxglossaries\space if you
4420     don't want any glossaries.)^^JThis document will not have a glossary}%
4421 }%

```

Suppress warning about no \makeglossaries

```

4422 \let\warn@nomakeglossaries\relax

```

Prevent user from also using \makeglossaries

```

4423 \let\makeglossaries\@no@makeglossaries

```

Allow sort key in printgloss family:

```

4424 \renewcommand*{\@printgloss@setsort}{%
4425   \let\@glo@assign@sortkey\@glo@assign@sortkey

```

Initialise default sort order:

```

4426   \def\@glo@sorttype{\@glo@default@sorttype}%
4427 }%

```

All entries must be defined in the preamble:

```

4428 \renewcommand*\new@glossaryentry[2]{%
4429   \PackageError{glossaries}{Glossary entries must be
4430     defined in the preamble^^Jwhen you use
4431     \string\makenoidxglossaries}%
4432   {Either move your definitions to the preamble or use
4433     \string\makeglossaries}%
4434 }%

```

```

Redefine \glsentrynumberlist
4435 \renewcommand*{\glsentrynumberlist}[1]{%
4436 \letcs{\@gls@loclist}{glo@\glsdetoklabel{##1}@loclist}%
4437 \ifdef\@gls@loclist
4438 {%
4439 \glsnoidxloclist{\@gls@loclist}%
4440 }%
4441 {%
4442 \ifglsentryexists{##1}%
4443 {%
4444 \GlossariesWarning{Missing location list for ‘##1’. Either
4445 a rerun is required or you haven’t referenced the entry.}%
4446 }%
4447 {%
4448 \PackageError{glossaries}{Glossary entry ‘##1’ has not been
4449 defined.}{}%
4450 }%
4451 }%
4452 }%

Redefine \glsdisplaynumberlist
4453 \renewcommand*{\glsdisplaynumberlist}[1]{%
4454 \letcs{\@gls@loclist}{glo@\glsdetoklabel{##1}@loclist}%
4455 \ifdef\@gls@loclist
4456 {%
4457 \def\@gls@noidxloclist@sep{%
4458 \def\@gls@noidxloclist@sep{%
4459 \def\@gls@noidxloclist@sep{%
4460 \glsnumlistsep
4461 }%
4462 \def\@gls@noidxloclist@finalsep{\glsnumlistlastsep}%
4463 }%
4464 }%
4465 \def\@gls@noidxloclist@finalsep{}%
4466 \def\@gls@noidxloclist@prev{}%
4467 \forlistloop{\glsnoidxdisplayloclisthandler}{\@gls@loclist}%
4468 \@gls@noidxloclist@finalsep
4469 \@gls@noidxloclist@prev
4470 }%
4471 {%
4472 ??\ifglsentryexists{##1}%
4473 {%
4474 \GlossariesWarning{Missing location list for ‘##1’. Either
4475 a rerun is required or you haven’t referenced the entry.}%
4476 }%
4477 {%
4478 \PackageError{glossaries}{Glossary entry ‘##1’ has not been
4479 defined.}{}%
4480 }%
4481 }%

```

4482 }%

Provide a generic way of iterating through the number list:

```

4483 \renewcommand*\glsnumberlistloop}[3]{%
4484   \letcs{\@gls@loclist}{gls@glstoklabel{##1}@loclist}%
4485   \let\@gls@org@glsnoidxdisplayloc\glsnoidxdisplayloc
4486   \let\@gls@org@glsseeformat\glsseeformat
4487   \let\glsnoidxdisplayloc##2\relax
4488   \let\glsseeformat##3\relax
4489   \ifdef\@gls@loclist
4490   {%
4491     \forlistloop{\glsnoidxnumberlistloophandler}{\@gls@loclist}%
4492   }%
4493   {%
4494     \ifglsentryexists{##1}%
4495     {%
4496       \GlossariesWarning{Missing location list for ‘##1’. Either
4497         a rerun is required or you haven’t referenced the entry.}%
4498     }%
4499     {%
4500       \PackageError{glossaries}{Glossary entry ‘##1’ has not been
4501         defined.}{}%
4502     }%
4503   }%
4504   \let\glsnoidxdisplayloc\@gls@org@glsnoidxdisplayloc
4505   \let\glsseeformat\@gls@org@glsseeformat
4506 }%
```

Modify sanitize sort function

```

4507 \let\@gls@sanitizesort\@gls@noidx@sanitizesort
4508 \let\@gls@nosanitizesort\@gls@noidx@nosanitizesort
4509 \@gls@noidx@setsanitizesort
4510 }
```

Preamble-only command:

```

4511 \onlypreamble{\makenoidxglossaries}
```

`\glsnumberlistloop` `\glsnumberlistloop{<label>}{<handler>}`

```

4512 \newcommand*\glsnumberlistloop}[2]{%
4513   \PackageError{glossaries}{\string\glsnumberlistloop\space
4514     only works with \string\makenoidxglossaries}{}%
4515 }
```

`numberlistloophandler` Handler macro for `\glsnumberlistloop`. (The argument should be in the form `\glsnoidxdisplayloc{<prefix>}{<counter>}{<format>}{<n>}`)

```

4516 \newcommand*\glsnoidxnumberlistloophandler}[1]{%
4517   #1%
4518 }
```

```

\@no@makeglossaries Can't use both \makeglossaries and \makenoidxglossaries
4519 \newcommand*{\@no@makeglossaries}{%
4520   \PackageError{glossaries}{You can't use both
4521     \string\makeglossaries\space and \string\makenoidxglossaries}%
4522   {Either use one or other (or none) of those commands but not both
4523     together.}%
4524 }

\@gls@noref@warn Warning when no instances of \@gls@reference found.
4525 \newcommand{\@gls@noref@warn}[1]{%
4526   \GlossariesWarning{\string\makenoidxglossaries\space
4527     is required to make \string\printnoidxglossary[type={#1}] work}%
4528 }

\gls@noidxglossary Write the glossary information to the aux file:
4529 \newcommand*{\gls@noidxglossary}{%
4530   \protected@write\@auxout{}{%
4531     \string\@gls@reference
4532     {\csname glo@\@gls@label @type\endcsname}%
4533     {\@gls@label}%
4534     {\string\glsnoidxdisplayloc
4535       {\@glo@counterprefix}%
4536       {\@gls@counter}%
4537       {\@glsnumberformat}%
4538       {\@glslocref}%
4539     }%
4540   }%
4541 }

```

### 1.13 Writing information to associated files

`\istfile` Deprecated.

```
4542 \def\istfile{\glswrite}
```

At the end of the document, the files should be created if `savewrites=true`.

```

4543 \AtEndDocument{%
4544   \glswritefiles
4545 }

```

`\@glswritefiles` Only write the files if `savewrites=true`

```

4546 \newcommand*{\@glswritefiles}{%
  Iterate through all the glossaries
4547   \forallglossaries{\@glo@type}{%
    Check for empty glossaries (patch provided by Patrick Häcker)
4548     \ifcsundef{glo@\@glo@type @filetok}%
4549     {%
4550       \def\gls@tmp{%

```

```

4551 }%
4552 {%
4553     \edef\gls@tmp{\expandafter\the
4554         \csname glo@\@glo@type @filetok\endcsname}%
4555 }%
4556 \ifx\gls@tmp\@empty
4557     \ifx\@glo@type\glsdefaulttype
4558         \GlossariesWarningNoLine{Glossary '@glo@type' has no
4559             entries.^^JRemember to use package option 'nomain' if
4560 you
4561             don't want to^^Juse the main glossary}%
4562     \else
4563         \GlossariesWarningNoLine{Glossary '@glo@type' has no
4564             entries}%
4565     \fi
4566 \else
4567     \@glsopenfile{\glswrite}{\@glo@type}%
4568     \immediate\write\glswrite{%
4569         \expandafter\the
4570         \csname glo@\@glo@type @filetok\endcsname}%
4571     \immediate\closeout\glswrite
4572 \fi
4573 }%
4574 }

```

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`

```

4575 \renewcommand*{\glossary}[1][\glsdefaulttype]{%
4576     \@glossary[#1]%
4577 }

```

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`

```

4578 \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`

```

4579 \newcommand{\@gls@renewglossary}{%
4580     \gdef\@glossary[##1]{\@bsphack\begingroup\@wrglossary{##1}}%

```

```

4581 \let\@gls@renewglossary\@empty
4582 }

```

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`

```

4583 \renewcommand*{\@wrglossary}[2]{%
4584   \ifglssavewrites
4585     \protected@edef\@gls@tmp{\the\csname glo@#1@filetok\endcsname#2}%
4586     \expandafter\global\expandafter\csname glo@#1@filetok\endcsname
4587       \expandafter{\@gls@tmp^^J}%
4588   \else
4589     \ifcsdef{glo@#1@file}%
4590     {%
4591       \expandafter\protected@write\csname glo@#1@file\endcsname{%
4592         \gls@disablepagerefexpansion}{#2}%
4593     }%
4594     {%
4595       \GlossariesWarning{No file defined for glossary ‘#1’}%
4596     }%
4597   \fi
4598 \endgroup\@esphack
4599 }

```

`\@do@wrglossary`

```

4600 \newcommand*{\@do@wrglossary}[1]{%
4601   \ifglsindexonlyfirst
4602     \ifglsused{#1}{\@do@wrglossary{#1}}%
4603   \else
4604     \@do@wrglossary{#1}%
4605   \fi
4606 }

```

`@protected@pagefmts` List of page formats to be protected against expansion.

```

4607 \newcommand{\gls@protected@pagefmts}{%
4608   \gls@numberpage,\gls@alphpage,\gls@Alphpage,\gls@romanpage,\gls@Romanpage%
4609 }

```

`blepagerefexpansion`

```

4610 \newcommand*{\gls@disablepagerefexpansion}{%
4611   \@for\@gls@this:=\gls@protected@pagefmts\do
4612   {%
4613     \expandafter\let\@gls@this\relax
4614   }%
4615 }

```

```

\gls@alphpage
4616 \newcommand*{\gls@alphpage}{\@alph\c@page}

\gls@Alphpage
4617 \newcommand*{\gls@Alphpage}{\@Alph\c@page}

\gls@numberpage
4618 \newcommand*{\gls@numberpage}{\number\c@page}

\gls@romanpage
4619 \newcommand*{\gls@romanpage}{\romannumeral\c@page}

\gls@Romanpage
4620 \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.
4621 \newcommand*{\@do@wrglossary}[1]{%
4622   \begingroup
      First a bit of hackery to prevent premature expansion of \c@page. Store original
      definitions:
4623   \let\orgthe\the
4624   \let\orgnumber\number
4625   \let\orgromannumeral\romannumeral
4626   \let\orgalph\@alph
4627   \let\orgAlph\@Alph
4628   \let\orgRoman\@Roman
      Redefine:
4629   \def\the##1{%
4630     \ifx##1\c@page \gls@numberpage\else\orgthe##1\fi}%
4631   \def\number##1{%
4632     \ifx##1\c@page \gls@numberpage\else\orgnumber##1\fi}%
4633   \def\romannumeral##1{%
4634     \ifx##1\c@page \gls@romanpage\else\orgromannumeral##1\fi}%
4635   \def\@Roman##1{%
4636     \ifx##1\c@page \gls@Romanpage\else\orgRoman##1\fi}%
4637   \def\@alph##1{%
4638     \ifx##1\c@page \gls@alphpage\else\orgalph##1\fi}%
4639   \def\@Alph##1{%
4640     \ifx##1\c@page \gls@Alphpage\else\orgAlph##1\fi}%
      Prevent expansion:
4641   \gls@disablepagerefexpansion
      Now store location in \glslocref:
4642   \protected@xdef\glslocref{\theglsentrycounter}%
4643   \endgroup

```

Escape any special characters

```
4644 \@gls@checkmkidxchars\@glslocref
```

Check if the hyper-location is the same as the location and set the hyper prefix.

```
4645 \expandafter\ifx\theHglentrycounter\theHglentrycounter\relax
4646 \def\@glo@counterprefix{}%
4647 \else
4648 \protected@edef\@glsHlocref{\theHglentrycounter}%
4649 \@gls@checkmkidxchars\@glsHlocref
4650 \edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix
4651 {\@glslocref}{\@glsHlocref}%
4652 }%
4653 \@do@gls@getcounterprefix
4654 \fi
```

De-tok label if required

```
4655 \edef\@gls@label{\glsdetoklabel{#1}}%
```

Write the information to file:

```
4656 \@do@wrglossary
4657 }
```

```
\@do@wrglossary
```

```
4658 \newcommand*\@do@wrglossary{%
```

Determine whether to use xindy or makeindex syntax

```
4659 \ifglxindy
```

Need to determine if the formatting information starts with a ( or ) indicating a range.

```
4660 \expandafter\@glo@check@mkidxrangechar\@glsnumberformat\@nil
4661 \def\@glo@range{}%
4662 \expandafter\if\@glo@prefix(\relax
4663 \def\@glo@range{:open-range}%
4664 \else
4665 \expandafter\if\@glo@prefix)\relax
4666 \def\@glo@range{:close-range}%
4667 \fi
4668 \fi
```

Write to the glossary file using xindy syntax.

```
4669 \glossary[\csname glo@\@gls@label @type\endcsname]{%
4670 (indexentry :tkey (\csname glo@\@gls@label @index\endcsname)

4671 :locref \string{"\@glo@counterprefix}{\@glslocref}\string" %
4672 :attr \string"\@gls@counter\@glo@suffix\string"
4673 \@glo@range
4674 )
4675 }%
4676 \else
```



Convert the format information into the format required for makeindex

```
4677 \set@glo@numformat{\@glo@numfmt}{\@gls@counter}{\@glsnumberformat}%
4678 {\@glo@counterprefix}%
```

Write to the glossary file using makeindex syntax.

```
4679 \glossary[\csname glo@\@gls@label @type\endcsname]{%
4680 \string\glossaryentry{\csname glo@\@gls@label @index\endcsname
4681 \@gls@encapchar\@glo@numfmt}{\@gls@locref}}%
4682 \fi
4683 }
```

`\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`.) NB this assumes that the prefix ends with a dot, which is the standard. (Otherwise it makes the xindy location classes more complicated.)

```
4684 \newcommand*\@gls@getcounterprefix[2]{%
4685 \edef\@gls@thisloc{#1}\edef\@gls@thisHloc{#2}%
4686 \ifx\@gls@thisloc\@gls@thisHloc
4687 \def\@glo@counterprefix{}%
4688 \else
4689 \def\@gls@get@counterprefix##1.#1##2\end@getprefix{%
4690 \def\@glo@tmp{##2}%
4691 \ifx\@glo@tmp\@empty
4692 \def\@glo@counterprefix{}%
4693 \else
4694 \def\@glo@counterprefix{##1}%
4695 \fi
4696 }%
4697 \@gls@get@counterprefix#2.#1\end@getprefix
```

Warn if no prefix can be formed.

```
4698 \ifx\@glo@counterprefix\@empty
4699 \GlossariesWarning{Hyper target ‘#2’ can’t be formed by
4700 prefixing~Jlocation ‘#1’. You need to modify the
4701 definition of \string\theH\@gls@counter~Jotherwise you
4702 will get the warning: “‘name{\@gls@counter.#1}’ has been~J
4703 referenced but does not exist”}%
4704 \fi
4705 \fi
4706 }
```

## 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.

```
4707 \newcommand{\@do@seeglossary}[2]{%
4708 \def\@gls@xref{#2}%
```

```

4709 \@onelevel@sanitize\@gls@xref
4710 \@gls@checkmkidxchars\@gls@xref
4711 \ifglxsindy
4712   \glossary[\csname glo@#1@type\endcsname]{%
4713     (indexentry
4714       :tkey (\csname glo@#1@index\endcsname)
4715       :xref (\string"\@gls@xref\string")
4716       :attr \string"see\string"
4717     )
4718   }%
4719 \else
4720   \glossary[\csname glo@#1@type\endcsname]{%
4721     \string\glossaryentry{\csname glo@#1@index\endcsname
4722       \@gls@encapchar glsseeformat\@gls@xref}{Z}}%
4723 \fi
4724 }

```

`\@gls@fixbraces` If no optional argument is specified, list needs to be enclosed in a set of braces.

```

4725 \def\@gls@fixbraces#1#2#3\@nil{%
4726   \ifx#2[\relax
4727     \@gls@fixbraces#1#2#3\@end@fixbraces
4728   \else
4729     \def#1{{#2#3}}%
4730   \fi
4731 }

```

`\@gls@fixbraces`

```

4732 \def\@gls@fixbraces#1[#2]#3\@end@fixbraces{%
4733   \def#1{[#2]{#3}}%
4734 }

```

`\glssee` `\glssee{<label>}{<cross-ref list>}`

```

4735 \DeclareRobustCommand*\glssee[3][\seename]{%
4736   \@do@seeglossary{#2}{[#1]{#3}}%
4737 \newcommand*\@glssee[3][\seename]{%
4738   \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.

```

4739 \DeclareRobustCommand*\glsseeformat[3][\seename]{%
4740   \emph{#1} \glsseelist{#2}}

```

`\glsseelist` `\glsseelist{<list>}` formats list of entry labels.

```

4741 \DeclareRobustCommand*\glsseelist[1]{%

```

If there is only one item in the list, set the last separator to do nothing.

```

4742   \let\@gls@dolast\relax

```

Don’t display separator on the first iteration of the loop

```

4743   \let\@gls@donext\relax

```

Iterate through the labels

```
4744 \@for\@gls@thislabel:=#1\do{%
```

Check if on last iteration of loop

```
4745 \ifx\@xfor@nextelement\@nnil
4746 \@gls@dolast
4747 \else
4748 \@gls@donext
4749 \fi
```

Display the entry for this label. (Expanding label as it's a temporary control sequence that's used elsewhere.)

```
4750 \expandafter\glsseeitem\expandafter{\@gls@thislabel}%
```

Update separators

```
4751 \let\@gls@dolast\glsseelastsep
4752 \let\@gls@donext\glsseesep
4753 }%
4754 }
```

`\glsseelastsep` Separator to use between penultimate and ultimate entries in a cross-referencing list.

```
4755 \newcommand*{\glsseelastsep}{\space\andname\space}
```

`\glsseesep` Separator to use between entries in a cross-referencing list.

```
4756 \newcommand*{\glsseesep}{, }
```

`\glsseeitem` `\glsseeitem{<label>}` formats individual entry in a cross-referencing list.

```
4757 \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.)

```
4758 \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.

```
4759 \newcommand*{\gls@save@numberlist}[1]{%
4760 \ifglssavenumberlist
4761 \toks@{#1}%
4762 \edef\@do@writeaux@info{%
4763 \noexpand\csgdef{glo@\glscurrententrylabel @numberlist}{\the\toks@}%
```

```

4764 }%
4765 \@onelevel@sanitize\@do@writeaux@info
4766 \protected@write\@auxout{}\@do@writeaux@info}%
4767 \fi
4768 }

```

`\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.)

```

4769 \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.

```

4770 \ifcsundef{printglossary}\%
4771 {%

```

If `\printglossary` is already defined, issue a warning and undefine it.

```

4772 \@gls@warnonglossdefined
4773 \undef\printglossary
4774 }

```

`\printglossary` has an optional argument. The default value is to set the glossary type to the main glossary.

```

4775 \newcommand*\printglossary[1][type=\glsdefaulttype]{%
4776 \@printglossary{#1}\@print@glossary}%
4777 }

```

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`

```

4778 \newcommand*\printglossaries{%
4779 \foralllglossaries{\@glo@type}\printglossary[type=\@glo@type]}%
4780 }

```

`\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.

```

4781 \newcommand*\printnoidxglossary[1][type=\glsdefaulttype]{%
4782 \@printglossary{#1}\@print@noidx@glossary}%
4783 }

```

`\printnoidxglossaries` Analogous to `\printglossaries`

```

4784 \newcommand*{\printnoidxglossaries}{%
4785   \forallglossaries{\@glo@type}{\printnoidxglossary[type=\@glo@type]}%
4786 }

```

`\@printgloss@setsort` Initialise to do nothing.

```

4787 \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.

```

4788 \newcommand{\@printglossary}[2]{%

```

Set up defaults.

```

4789   \def\@glo@type{\glsdefaulttype}%
4790   \def\glossarytitle{\csname @glo@type\@glo@type @title\endcsname}%

```

```

4791   \def\glossarytoctitle{\glossarytitle}%
4792   \let\org@glossarytitle\glossarytitle
4793   \def\@glossarystyle{}%
4794   \def\gls@dotocitle{\glssettocitle{\@glo@type}}%

```

Store current value of `\glossaryentrynumbers`. (This may be changed via the optional argument)

```

4795   \let\@org@glossaryentrynumbers\glossaryentrynumbers

```

Localise the effects of the optional argument

```

4796   \bgroup

```

Activate or deactivate sort key:

```

4797   \@printgloss@setsort

```

Determine settings specified in the optional argument.

```

4798   \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)

```

4799   \ifx\glossarytitle\org@glossarytitle
4800   \else
4801     \expandafter\let\csname @glo@type\@glo@type @title\endcsname
4802       \glossarytitle
4803   \fi

```

Allow a high-level user command to indicate the current glossary

```

4804   \let\currentglossary\@glo@type

```

Enable individual number lists to be suppressed.

```

4805   \let\org@glossaryentrynumbers\glossaryentrynumbers
4806   \let\glsnonextpages\@glsnonextpages

```

Enable individual number list to be activated:

```

4807   \let\glsnextpages\@glsnextpages

```

Enable suppression of description terminators.

```
4808 \let\nopostdesc\@nopostdesc
```

Set up the entry for the TOC

```
4809 \gls@dotoc@title
```

Set the glossary style

```
4810 \@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):

```
4811 \let\gls@org@glossaryentryfield\glossentry
4812 \let\gls@org@glossarysubentryfield\subglossentry
4813 \renewcommand{\glossentry}[1]{%
4814   \xdef\glscurrententrylabel{\glsdetoklabel{##1}}%
4815   \gls@org@glossaryentryfield{##1}%
4816 }%
4817 \renewcommand{\subglossentry}[2]{%
4818   \xdef\glscurrententrylabel{\glsdetoklabel{##2}}%
4819   \gls@org@glossarysubentryfield{##1}{##2}%
4820 }%
```

Now do the handler macro that deals with the actual glossary:

```
4821 #2%
```

End the current scope

```
4822 \egroup
```

Reset `\glossaryentrynumbers`

```
4823 \global\let\glossaryentrynumbers\@org@glossaryentrynumbers
```

Suppress warning about no `\printglossary`

```
4824 \global\let\warn@noprintglossary\relax
4825 }
```

`\@print@glossary` Internal workings of `\printglossary` dealing with reading the external file.

```
4826 \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.)

```
4827 \makeatletter
```

Input the glossary file, if it exists.

```
4828 \@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.

```
4829 \IfFileExists{\jobname.\csname @glo@type\@glo@type @in\endcsname}%
4830 {}%
4831 {\null}%
```

If xindy is being used, need to write the language dependent information to the .aux file for makeglossaries.

```

4832 \ifglsxindy
4833 \ifcsundef{@xdy@\@glo@type @language}%
4834 {%
4835 \edef\@do@auxoutstuff{%
4836 \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.

```

4837 \noexpand\immediate\noexpand\write\@auxout{%
4838 \string\providecommand\string\@xdylanguage[2]{}%
4839 \noexpand\immediate\noexpand\write\@auxout{%
4840 \string\@xdylanguage{\@glo@type}{\@xdy@main@language}}%
4841 }%
4842 }%
4843 }%
4844 {%
4845 \edef\@do@auxoutstuff{%
4846 \noexpand\AtEndDocument{%
4847 \noexpand\immediate\noexpand\write\@auxout{%
4848 \string\providecommand\string\@xdylanguage[2]{}%
4849 \noexpand\immediate\noexpand\write\@auxout{%
4850 \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type
4851 @language\endcsname}}%
4852 }%
4853 }%
4854 }%
4855 \@do@auxoutstuff
4856 \edef\@do@auxoutstuff{%
4857 \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.

```

4858 \noexpand\immediate\noexpand\write\@auxout{%
4859 \string\providecommand\string\@gls@codepage[2]{}%
4860 \noexpand\immediate\noexpand\write\@auxout{%
4861 \string\@gls@codepage{\@glo@type}{\@gls@codepage}}%
4862 }%
4863 }%
4864 \@do@auxoutstuff
4865 \fi

```

Activate warning if \makeglossaries hasn't been used.

```

4866 \renewcommand*{\@warn@nomakeglossaries}{%
4867 \GlossariesWarningNoLine{\string\makeglossaries\space
4868 hasn't been used,^^Jthe glossaries will not be updated}%
4869 }%

```

4870 }

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`

```
4871 \newcommand*{\@glo@sortentries}[2]{%
4872   \def\@glo@sortinglist{}%
4873   \def\@glo@sortinghandler{#1}%
4874   \edef\@glo@type{#2}%
4875   \forlistcsloop{\@glo@do@sortentries}{\@glsref@#2}%
4876   \csdef{\@glsref@#2}{}%
4877   \@for\@this@label:=\@glo@sortinglist\do{%
```

Has this entry already been added?

```
4878     \xifinlistcs{\@this@label}{\@glsref@#2}%
4879     {}%
4880     {%
4881       \listcsxadd{\@glsref@#2}{\@this@label}%
4882     }%
4883     \ifcsdef{\@glo@sortingchildren@ \@this@label}%
4884     {%
4885       \@glo@addchildren{#2}{\@this@label}%
4886     }%
4887     {}%
4888   }%
4889 }
```

`\@glo@addchildren`

`\@glo@addchildren{<type>}{<parent>}`

```
4890 \newcommand*{\@glo@addchildren}[2]{%
```

Scope to allow nesting.

```
4891   \bgroup
4892   \letcs{\@glo@childlist}{\@glo@sortingchildren@#2}%
4893   \@for\@this@childlabel:=\@glo@childlist\do
4894   {%
```

Check this label hasn't already been added.

```
4895     \xifinlistcs{\@this@childlabel}{\@glsref@#1}%
4896     {}%
4897     {%
4898       \listcsxadd{\@glsref@#1}{\@this@childlabel}%
4899     }%
```



Does this child have children?

```

4900      \ifcsdef{@glo@sortingchildren@ \@this@childlabel}%
4901      {%
4902      \@glo@addchildren{#1}{\@this@childlabel}%
4903      }%
4904      {%
4905      }%
4906      }%
4907      \egroup
4908 }

```

@glo@do@sortentries

```

4909 \newcommand*{\@glo@do@sortentries}[1]{%
4910   \ifglshasparent{#1}%
4911   {%

```

This entry has a parent, so add it to the child list

```

4912     \edef\@glo@parent{\csuse{glo@glsetoklabel{#1}@parent}}%
4913     \ifcsundef{@glo@sortingchildren@\@glo@parent}%
4914     {%
4915     \csdef{@glo@sortingchildren@\@glo@parent}{}%
4916     }%
4917     {}%
4918     \expandafter\@glo@sortedinsert
4919     \csname @glo@sortingchildren@\@glo@parent\endcsname{#1}%

```

Has the parent been added?

```

4920     \xifinlistcs{\@glo@parent}{\glstrref@\@glo@type}%
4921     {%

```

Yes, it has so do nothing.

```

4922     }%
4923     {%

```

No, it hasn't so add it now.

```

4924     \expandafter\@glo@do@sortentries\expandafter{\@glo@parent}%
4925     }%
4926     }%
4927     {%
4928     \@glo@sortedinsert{\@glo@sortinglist}{#1}%
4929     }%
4930 }

```

\@glo@sortedinsert

\@glo@sortedinsert{<list>}{<entry label>}
---

Insert into list.

```

4931 \newcommand*{\@glo@sortedinsert}[2]{%
4932   \dtlinsertinto{#2}{#1}{\@glo@sortinghandler}%
4933 }%

```

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`

```

4934 \newcommand*{\@glo@sorthandler@word}[2]{%
4935   \letcs\@gls@sort@A{glo@\glsdetoklabel{\#1}@sort}%
4936   \letcs\@gls@sort@B{glo@\glsdetoklabel{\#2}@sort}%
4937   \edef\glo@do@compare{%
4938     \noexpand\dtlwordindexcompare{\noexpand\dtl@sortresult}%
4939     {\expandonce\@gls@sort@B}%
4940     {\expandonce\@gls@sort@A}%
4941   }%
4942   \glo@do@compare
4943 }
```

`@sorthandler@letter`

```

4944 \newcommand*{\@glo@sorthandler@letter}[2]{%
4945   \letcs\@gls@sort@A{glo@\glsdetoklabel{\#1}@sort}%
4946   \letcs\@gls@sort@B{glo@\glsdetoklabel{\#2}@sort}%
4947   \edef\glo@do@compare{%
4948     \noexpand\dtlletterindexcompare{\noexpand\dtl@sortresult}%
4949     {\expandonce\@gls@sort@B}%
4950     {\expandonce\@gls@sort@A}%
4951   }%
4952   \glo@do@compare
4953 }
```

`\lo@sorthandler@case` Case-sensitive sort.

```

4954 \newcommand*{\@glo@sorthandler@case}[2]{%
4955   \letcs\@gls@sort@A{glo@\glsdetoklabel{\#1}@sort}%
4956   \letcs\@gls@sort@B{glo@\glsdetoklabel{\#2}@sort}%
4957   \edef\glo@do@compare{%
4958     \noexpand\dtlcompare{\noexpand\dtl@sortresult}%
4959     {\expandonce\@gls@sort@B}%
4960     {\expandonce\@gls@sort@A}%
4961   }%
4962   \glo@do@compare
4963 }
```

`@sorthandler@nocase` Case-insensitive sort.

```

4964 \newcommand*{\@glo@sorthandler@nocase}[2]{%
4965   \letcs\@gls@sort@A{glo@\glsdetoklabel{\#1}@sort}%
4966   \letcs\@gls@sort@B{glo@\glsdetoklabel{\#2}@sort}%
4967   \edef\glo@do@compare{%
4968     \noexpand\dtlicompare{\noexpand\dtl@sortresult}%
4969     {\expandonce\@gls@sort@B}%
4970     {\expandonce\@gls@sort@A}%
4971   }%
```

```

4972 \glo@do@compare
4973 }

@glo@sortmacro@word Sort macro for ‘word’
4974 \newcommand*{\@glo@sortmacro@word}[1]{%
4975 \ifdefstring{\@glo@default@sorttype}{standard}%
4976 {%
4977 \@glo@sortentries{\@glo@sorthandler@word}{#1}%
4978 }%
4979 {%
4980 \PackageError{glossaries}{Conflicting sort options:^^J
4981 \string\usepackage[sort=\@glo@default@sorttype]{glossaries}^^J
4982 \string\printnoidxglossary[sort=word]}{}%
4983 }%
4984 }

@glo@sortmacro@letter Sort macro for ‘letter’
4985 \newcommand*{\@glo@sortmacro@letter}[1]{%
4986 \ifdefstring{\@glo@default@sorttype}{standard}%
4987 {%
4988 \@glo@sortentries{\@glo@sorthandler@letter}{#1}%
4989 }%
4990 {%
4991 \PackageError{glossaries}{Conflicting sort options:^^J
4992 \string\usepackage[sort=\@glo@default@sorttype]{glossaries}^^J
4993 \string\printnoidxglossary[sort=letter]}{}%
4994 }%
4995 }

@glo@sortmacro@standard Sort macro for ‘standard’. (Use either ‘word’ or ‘letter’ order.)
4996 \newcommand*{\@glo@sortmacro@standard}[1]{%
4997 \ifdefstring{\@glo@default@sorttype}{standard}%
4998 {%
4999 \ifcsdef{\@glo@sorthandler@\glsorder}%
5000 {%
5001 \@glo@sortentries{\csuse{\@glo@sorthandler@\glsorder}}{#1}%
5002 }%
5003 {%
5004 \PackageError{glossaries}{Unknown sort handler ‘\glsorder’}{}%
5005 }%
5006 }%
5007 {%
5008 \PackageError{glossaries}{Conflicting sort options:^^J
5009 \string\usepackage[sort=\@glo@default@sorttype]{glossaries}^^J
5010 \string\printnoidxglossary[sort=standard]}{}%
5011 }%
5012 }

@glo@sortmacro@case Sort macro for ‘case’

```

```

5013 \newcommand*{\@glo@sortmacro@case}[1]{%
5014   \ifdefstring{\@glo@default@sorttype}{standard}%
5015   {%
5016     \@glo@sortentries{\@glo@sorthandler@case}{#1}%
5017   }%
5018   {%
5019     \PackageError{glossaries}{Conflicting sort options:^^J
5020       \string\usepackage[sort=\@glo@default@sorttype]{glossaries}^^J
5021       \string\printnoidxglossary[sort=case]}{}%
5022   }%
5023 }

```

\lo@sortmacro@nocase Sort macro for ‘nocase’

```

5024 \newcommand*{\@glo@sortmacro@nocase}[1]{%
5025   \ifdefstring{\@glo@default@sorttype}{standard}%
5026   {%
5027     \@glo@sortentries{\@glo@sorthandler@nocase}{#1}%
5028   }%
5029   {%
5030     \PackageError{glossaries}{Conflicting sort options:^^J
5031       \string\usepackage[sort=\@glo@default@sorttype]{glossaries}^^J
5032       \string\printnoidxglossary[sort=nocase]}{}%
5033   }%
5034 }

```

\@glo@sortmacro@def Sort macro for ‘def’. The order of definition is given in \glolist@<type>.

```

5035 \newcommand*{\@glo@sortmacro@def}[1]{%
5036   \def\@glo@sortinglist{}%
5037   \forlslentries[#1]{\@gls@thislabel}%
5038   {%
5039     \xifinlistcs{\@gls@thislabel}{\@glsref@#1}%
5040     {%
5041       \listeadadd{\@glo@sortinglist}{\@gls@thislabel}%
5042     }%
5043   }%

```

Hasn’t been referenced.

```

5044   }%
5045 }%
5046 \cslet{\@glsref@#1}{\@glo@sortinglist}%
5047 }

```

\lo@sortmacro@def@do This won’t include parent entries that haven’t been referenced.

```

5048 \newcommand*{\@glo@sortmacro@def@do}[1]{%
5049   \ifinlistcs{#1}{\@glsref@\@glo@type}%
5050   {}%
5051   {%
5052     \listcsadd{\@glsref@\@glo@type}{#1}%
5053   }%

```

```

5054 \ifcsdef{@glo@sortingchildren@#1}%
5055 {%
5056 \@glo@addchildren{\@glo@type}{#1}%
5057 }%
5058 {}%
5059 }

\@glo@sortmacro@use Sort macro for 'use'. (No sorting is required, as the entries are already in order
of use, so do nothing.)
5060 \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.
5061 \newcommand*{\@print@noidx@glossary}{%
5062 \ifcsdef{@glsref@\@glo@type}%
5063 {%
Sort the entries:
5064 \ifcsdef{@glo@sortmacro@\@glo@sorttype}%
5065 {%
5066 \csuse{@glo@sortmacro@\@glo@sorttype}{\@glo@type}%
5067 }%
5068 {%
5069 \PackageError{glossaries}{Unknown sort handler '@@glo@sorttype'}{}%
5070 }%
Do the glossary heading and preamble
5071 \glossarysection[\glossarytoctitle]{\glossarytitle}%
5072 \glossarypreamble
5073 \begin{theglossary}%
5074 \glossaryheader
5075 \glsresetentrylist
5076 \def\@gls@currentlettergroup{}%
Iterate through the entries.
5077 \forlistcsloop{\@gls@noidx@do}{\@glsref@\@glo@type}%
Finally end the glossary and do the postamble:
5078 \end{theglossary}%
5079 \glossarypostamble
5080 }%
5081 {}%
5082 \@gls@noref@warn{\@glo@type}%
5083 }%
5084 }

```

\glo@grabfirst

```
5085 \def\glo@grabfirst#1#2\@nil{%
5086   \def\@gls@firsttok{#1}%
5087   \ifdefempty\@gls@firsttok
5088   {%
5089     \def\@glo@thislettergrp{0}%
5090   }%
5091   {%
5092 % Sanitize it:
5093 %   \begin{macrocode}
5094   \@onelevel@sanitize\@gls@firsttok
5095   Fetch the first letter:
5096   \expandafter\@glo@grabfirst\@gls@firsttok{}\}\@nil
5097 }
```

\@glo@grabfirst

```
5098 \def\@glo@grabfirst#1#2\@nil{%
5099   \ifdefempty\@glo@thislettergrp
5100   {%
5101     \def\@glo@thislettergrp{glssymbols}%
5102   }%
5103   {%
5104     \count@=\uccode'#1\relax
5105     \ifnum\count@=0\relax
5106       \def\@glo@thislettergrp{glssymbols}%
5107     \else
5108       \ifdefstring\@glo@sorttype{case}%
5109       {%
5110         \count@='#1\relax
5111       }%
5112       {%
5113       }%
5114       \edef\@glo@thislettergrp{\the\count@}%
5115     \fi
5116   }%
5117 }
```

\@gls@noidx@do    Handler for list iteration used by \@print@noidx@glossary. The argument is the entry label. This only allows one sublevel.

```
5118 \newcommand{\@gls@noidx@do}[1]{%
5119   Get this entry's location list
5120   \letcs{\@gls@loclist}{glo@glsdetoklabel{#1}@loclist}%
5121   Does this entry have a parent?
5122   \ifglshasparent{#1}%
5123   {%
```

Has a parent.

```

5122 \gls@level=\csuse{glo@glsdetoklabel{#1}@level}\relax
5123 \ifdefvoid{\@gls@loclist}
5124 {%
5125   \subglossentry{\gls@level}{#1}{}%
5126 }%
5127 {%
5128   \subglossentry{\gls@level}{#1}%
5129   {%
5130     \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
5131   }%
5132 }%
5133 }%
5134 {%

```

Doesn't have a parent Get this entry's sort key

```

5135 \letcs{\@gls@sort}{glo@glsdetoklabel{#1}@sort}%

```

Fetch the first letter:

```

5136 \expandafter\glo@grabfirst\@gls@sort{}{}\@nil
5137 \ifdefequal{\@glo@thislettergrp}{\@gls@currentlettergroup}%
5138 {}%
5139 {%

```

Do the group header:

```

5140 \ifdefempty{\@gls@currentlettergroup}{\@gls@groupskip}%
5141 \gls@groupheading{\@glo@thislettergrp}%
5142 }%
5143 \let\@gls@currentlettergroup\@glo@thislettergrp

```

Do this entry:

```

5144 \ifdefvoid{\@gls@loclist}
5145 {%
5146   \glossentry{#1}{}%
5147 }%
5148 {%
5149   \glossentry{#1}%
5150   {%
5151     \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
5152   }%
5153 }%
5154 }%
5155 }

```

\glsnoidxloclist    \glsnoidxloclist{<list cs>}

Display location list.

```

5156 \newcommand*{\glsnoidxloclist}[1]{%
5157   \def\@gls@noidxloclist@sep{%

```

```

5158 \def\@gls@noidxloclist@prev{}%
5159 \forlistloop{\glsnoidxloclisthandler}{#1}%
5160 }

```

`noidxloclisthandler` Handler for location list iterator.

```

5161 \newcommand*\glsnoidxloclisthandler[1]{%
5162 \ifdefstring{\@gls@noidxloclist@prev}{#1}%
5163 {%
    Same as previous location so skip.
5164 }%
5165 {%
5166 \@gls@noidxloclist@sep
5167 #1%
5168 \def\@gls@noidxloclist@sep{\delimN}%
5169 \def\@gls@noidxloclist@prev{#1}%
5170 }%
5171 }

```

`splayloclisthandler` Handler for location list iterator when used with `\glsdisplaynumberlist`.

```

5172 \newcommand*\glsnoidxdisplayloclisthandler[1]{%
5173 \ifdefstring{\@gls@noidxloclist@prev}{#1}%
5174 {%
    Same as previous location so skip.
5175 }%
5176 {%
5177 \@gls@noidxloclist@sep
5178 \@gls@noidxloclist@prev
5179 \def\@gls@noidxloclist@prev{#1}%
5180 }%
5181 }

```

`\glsnoidxdisplayloc` `\glsnoidxdisplayloc{<prefix>}{<counter>}{<format>}{<location>}`

Display a location in the location list.

```

5182 \newcommand*\glsnoidxdisplayloc[4]{%
5183 \setentrycounter[#1]{#2}%
5184 \csuse{#3}{#4}%
5185 }

```

`\@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.

```

5186 \newcommand*\@gls@reference[3]{%

```



Add to label list

```

5187 \glsdoifexistsorwarn{#2}%
5188 {%
5189   \ifcsundef{@glsref@#1}{\csgdef{@glsref@#1}{}}{}%
5190   \ifinlistcs{#2}{@glsref@#1}%
5191   {}%
5192   {\listcsgadd{@glsref@#1}{#2}}%

```

Add to location list

```

5193 \ifcsundef{glo@glsdetoklabel{#2}@loclist}%
5194 {\csgdef{glo@glsdetoklabel{#2}@loclist}{}}%
5195 {}%
5196 \listcsgadd{glo@glsdetoklabel{#2}@loclist}{#3}%
5197 }%
5198 }

```

The keys that can be used in the optional argument to `\printglossary` or `\printnoidxglossary` are as follows: The `type` key sets the glossary type.

```

5199 \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`.

```

5200 \define@key{printgloss}{title}{%
5201   \def@glossarytitle{#1}%
5202   \let@gls@dotoc@title\relax
5203 }

```

The `toctitle` sets the text used for the relevant entry in the table of contents.

```

5204 \define@key{printgloss}{toctitle}{%
5205   \def@glossarytoctitle{#1}%
5206   \let@gls@dotoc@title\relax
5207 }

```

The `style` key sets the glossary style (but only for the given glossary).

```

5208 \define@key{printgloss}{style}{%
5209   \ifcsundef{@glsstyle@#1}%
5210   {%
5211     \PackageError{glossaries}%
5212     {Glossary style ‘#1’ undefined}{}%
5213   }%
5214   {%
5215     \def@glossarystyle{\setglossentrycompatibility
5216       \csname @glsstyle@#1\endcsname}%
5217   }%
5218 }

```

The `numberedsection` key determines if this glossary should be in a numbered section.

```

5219 \define@choicekey{printgloss}{numberedsection}[\val\nr]{%
5220 false,nolabel,autolabel,nameref}[nolabel]{%
5221   \ifcase\nr\relax

```

```

5222 \renewcommand*{\@@glossarysecstar}{*}%
5223 \renewcommand*{\@@glossaryseclabel}{}%
5224 \or
5225 \renewcommand*{\@@glossarysecstar}{}%
5226 \renewcommand*{\@@glossaryseclabel}{}%
5227 \or
5228 \renewcommand*{\@@glossarysecstar}{}%
5229 \renewcommand*{\@@glossaryseclabel}{\label{\glsautoprefix\@glo@type}}%
5230 \or
5231 \renewcommand*{\@@glossarysecstar}{*}%
5232 \renewcommand*{\@@glossaryseclabel}{%
5233 \protected@edef\@currentlabelname{\glossarytoctitle}%
5234 \label{\glsautoprefix\@glo@type}}%
5235 \fi
5236 }

```

The `nogroupskip` key determines whether or not there should be a vertical gap between glossary groups.

```

5237 \define@choicekey{printgloss}{nogroupskip}{true,false}[true]{%
5238 \csuse{glsnogroupskip#1}%
5239 }

```

The `nonumberlist` key determines if this glossary should have a number list.

```

5240 \define@boolkey{printgloss}[gls]{nonumberlist}[true]{%
5241 \ifglsnonumberlist
5242 \def\glossaryentrynumbers##1{%
5243 \else
5244 \def\glossaryentrynumbers##1{##1}%
5245 \fi}

```

The `sort` key sets the glossary sort handler (`\printnoidxglossary` only).

```

5246 \define@key{printgloss}{sort}{\@glo@assign@sortkey{#1}}

```

`\@no@assign@sortkey` Issue error if used with `\printglossary`

```

5247 \newcommand*{\@glo@no@assign@sortkey}[1]{%
5248 \PackageError{glossaries}{‘sort’ key not permitted with
5249 \string\printglossary}%
5250 {The ‘sort’ key may only be used with \string\printnoidxglossary}%
5251 }

```

`\@glo@assign@sortkey` For use with `\printnoidxglossary`

```

5252 \newcommand*{\@glo@assign@sortkey}[1]{%
5253 \def\@glo@sorttype{#1}%
5254 }

```

`\@glsnonextpages` Suppresses the next number list only. Global assignments required as it may not occur in the same level of grouping as the next numberlist. (For example, if `\glsnonextpages` is 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.

```

5255 \newcommand*{\@glsnonextpages}{%
5256   \gdef\glossaryentrynumbers##1{%
5257     \glsresetentrylist
5258   }%
5259 }

```

`\@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.

```

5260 \newcommand*{\@glsnextpages}{%
5261   \gdef\glossaryentrynumbers##1{%
5262     ##1\glsresetentrylist}}

```

`\glsresetentrylist` Resets `\glossaryentrynumbers`

```

5263 \newcommand*{\glsresetentrylist}{%
5264   \global\let\glossaryentrynumbers\org@glossaryentrynumbers}

```

`\glsnonextpages` Outside of `\printglossary` this does nothing.

```

5265 \newcommand*{\glsnonextpages}{}

```

`\glsnextpages` Outside of `\printglossary` this does nothing.

```

5266 \newcommand*{\glsnextpages}{}

```

`glossaryentry` If the `entrycounter` package option has been used, define a counter to number each level 0 entry.

```

5267 \ifglentrycounter
5268   \ifx\@gls@counterwithin\@empty
5269     \newcounter{glossaryentry}
5270   \else
5271     \newcounter{glossaryentry}[\@gls@counterwithin]
5272   \fi
5273   \def\theHglossaryentry{\currentglossary.\theglossaryentry}
5274 \fi

```

`glossarysubentry` If the `subentrycounter` package option has been used, define a counter to number each level 1 entry.

```

5275 \ifglssubentrycounter
5276   \ifglentrycounter
5277     \newcounter{glossarysubentry}[glossaryentry]
5278   \else
5279     \newcounter{glossarysubentry}
5280   \fi
5281   \def\theHglossarysubentry{\currentglssubentry.\theglossarysubentry}
5282 \fi

```

`\resetsubentrycounter` Resets the glossarysubentry counter.

```
5283 \ifglssubentrycounter
5284   \newcommand*{\glsresetsubentrycounter}{%
5285     \setcounter{glossarysubentry}{0}%
5286   }
5287 \else
5288   \newcommand*{\glsresetsubentrycounter}{}
5289 \fi
```

`\resetsubentrycounter` Resets the glossareentry counter.

```
5290 \ifglseentrycounter
5291   \newcommand*{\glsresetentrycounter}{%
5292     \setcounter{glossaryentry}{0}%
5293   }
5294 \else
5295   \newcommand*{\glsresetentrycounter}{}
5296 \fi
```

`\glsstepentry` Advance the glossaryentry counter if in use. The argument is the label associated with the entry.

```
5297 \ifglseentrycounter
5298   \newcommand*{\glsstepentry}[1]{%
5299     \refstepcounter{glossaryentry}%
5300     \label{glsentry-\glsdetoklabel{#1}}%
5301   }
5302 \else
5303   \newcommand*{\glsstepentry}[1]{}
5304 \fi
```

`\glsstepsubentry` Advance the glossarysubentry counter if in use. The argument is the label associated with the subentry.

```
5305 \ifglssubentrycounter
5306   \newcommand*{\glsstepsubentry}[1]{%
5307     \edef\currentglssubentry{\glsdetoklabel{#1}}%
5308     \refstepcounter{glossarysubentry}%
5309     \label{glsentry-\currentglssubentry}%
5310   }
5311 \else
5312   \newcommand*{\glsstepsubentry}[1]{}
5313 \fi
```

`\glsrefentry` Reference the entry or sub-entry counter if in use, otherwise just do `\gls`.

```
5314 \ifglseentrycounter
5315   \newcommand*{\glsrefentry}[1]{\ref{glsentry-\glsdetoklabel{#1}}}
5316 \else
5317   \ifglssubentrycounter
5318     \newcommand*{\glsrefentry}[1]{\ref{glsentry-\glsdetoklabel{#1}}}
5319   \else
```

```

5320 \newcommand*{\glsrefentry}[1]{\gls{#1}}
5321 \fi
5322 \fi

```

`\glsentrycounterlabel` Defines how to display the glossaryentry counter.

```

5323 \ifglentrycounter
5324 \newcommand*{\glsentrycounterlabel}{\theglossaryentry.\space}
5325 \else
5326 \newcommand*{\glsentrycounterlabel}{}
5327 \fi

```

`\glsentrysubcounterlabel` Defines how to display the glossarysubentry counter.

```

5328 \ifglssubentrycounter
5329 \newcommand*{\glsentrysubcounterlabel}{\theglossarysubentry)\space}
5330 \else
5331 \newcommand*{\glsentrysubcounterlabel}{}
5332 \fi

```

`\glsentryitem` Step and display glossaryentry counter, if appropriate.

```

5333 \ifglentrycounter
5334 \newcommand*{\glsentryitem}[1]{%
5335 \glsstepentry{#1}\glsentrycounterlabel
5336 }
5337 \else
5338 \newcommand*{\glsentryitem}[1]{\glsresetsubentrycounter}
5339 \fi

```

`\glssubentryitem` Step and display glossarysubentry counter, if appropriate.

```

5340 \ifglssubentrycounter
5341 \newcommand*{\glssubentryitem}[1]{%
5342 \glsstepsubentry{#1}\glssubentrycounterlabel
5343 }
5344 \else
5345 \newcommand*{\glssubentryitem}[1]{}
5346 \fi

```

`\theglossary` If the `\theglossary` environment has already been defined, a warning will be issued. This environment should be redefined by glossary styles.

```

5347 \ifcsundef{theglossary}%
5348 {%
5349 \newenvironment{theglossary}{}{}}%
5350}%
5351 {%
5352 \@gls@warnontheGLOSSdefined
5353 \renewenvironment{theglossary}{}{}}%
5354 }

```

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

```
5355 \newcommand*{\glossaryheader}{}

```

\glstarget \glstarget{<label>}{<name>}

Provide user interface to \@glstarget to make it easier to modify the glossary style in the document.

```
5356 \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>}

```
5357 \providecommand*{\compatibleglossentry}[2]{%
5358   \toks@{#2}%
5359   \protected@edef\@do@glossentry{\noexpand\glossaryentryfield{#1}%
5360     {\noexpand\glsnamefont
5361       {\expandafter\expandonce\csname glo@#1@name\endcsname}}}%
5362     {\expandafter\expandonce\csname glo@#1@desc\endcsname}%
5363     {\expandafter\expandonce\csname glo@#1@symbol\endcsname}%
5364     {\the\toks@}}%
5365   }%
5366   \@do@glossentry
5367 }
```

\glossentryname

```
5368 \newcommand*{\glossentryname}[1]{%
5369   \glsdoifexistsorwarn{#1}%
5370   {%
5371     \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
5372     \expandafter\glsnamefont\expandafter{\glo@name}%
5373   }%
5374 }
```

\Glossentryname

```
5375 \newcommand*{\Glossentryname}[1]{%
5376   \glsdoifexistsorwarn{#1}%
5377   {%
5378     \glsnamefont{\Glsentryname{#1}}%

```

```

5379 }%
5380 }

```

`\glossentrydesc`

```

5381 \newcommand*{\glossentrydesc}[1]{%
5382   \glsdoifexistsorwarn{#1}%
5383   {%
5384     \glsentrydesc{#1}%
5385   }%
5386 }

```

`\Glossentrydesc`

```

5387 \newcommand*{\Glossentrydesc}[1]{%
5388   \glsdoifexistsorwarn{#1}%
5389   {%
5390     \Glsentrydesc{#1}%
5391   }%
5392 }

```

`\glossentrysymbol`

```

5393 \newcommand*{\glossentrysymbol}[1]{%
5394   \glsdoifexistsorwarn{#1}%
5395   {%
5396     \glsentrysymbol{#1}%
5397   }%
5398 }

```

`\Glossentrysymbol`

```

5399 \newcommand*{\Glossentrysymbol}[1]{%
5400   \glsdoifexistsorwarn{#1}%
5401   {%
5402     \Glsentrysymbol{#1}%
5403   }%
5404 }

```

`compatiblesubglossentry` `\subglossentry{<level>}{<label>}{<page-list>}`

```

5405 \providecommand*{\compatiblesubglossentry}[3]{%
5406   \toks@{#3}%
5407   \protected@edef\@do@subglossentry{\noexpand\glossarysubentryfield{\number#1}%
5408     {#2}%
5409     {\noexpand\glsnamefont
5410       {\expandafter\expandonce\csname glo@#2@name\endcsname}}}%
5411     {\expandafter\expandonce\csname glo@#2@desc\endcsname}%
5412     {\expandafter\expandonce\csname glo@#2@symbol\endcsname}%
5413     {\the\toks@}}%
5414   }%

```

```

5415 \@do@subglossentry
5416 }

```

sentrycompatibility

```

5417 \newcommand*{\setglossentrycompatibility}{%
5418   \let\glossentry\compatibleglossentry
5419   \let\subglossentry\compatiblesubglossentry
5420 }
5421 \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.

```

5422 \newcommand{\glossaryentryfield}[5]{%
5423   \GlossariesWarning
5424   {Deprecated use of \string\glossaryentryfield.^^J
5425     I recommend you change to \string\glossentry.^^J
5426     If you've just upgraded, try removing your gls auxiliary
5427     files^^J and recompile}%
5428   \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.)

```

5429 \newcommand*{\glossarysubentryfield}[6]{%
5430   \GlossariesWarning
5431   {Deprecated use of \string\glossarysubentryfield.^^J
5432     I recommend you change to \string\subglossentry.^^J
5433     If you've just upgraded, try removing your gls auxiliary
5434     files^^J and recompile}%
5435   \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`

```
5436 \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`

```
5437 \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`

```
5438 \newcommand*{\glsgetgrouptitle}[1]{%
5439   \@gls@getgrouptitle{#1}{\@gls@grptitle}%
5440   \@gls@grptitle
5441 }
```

`\@gls@getgrouptitle` Gets the group title specified by the label (first argument) and stores in the second argument, which must be a control sequence.

```
5442 \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.

```

5443 \dtl@ifsingle{#1}%
5444 {%
5445   \ifcsundef{#1groupname}{\def#2{#1}}{\letcs#2{#1groupname}}%
5446 }%
5447 {%
5448   \ifboolexpr{test{\ifstrequal{#1}{glssymbols}}
5449               or test{\ifstrequal{#1}{glsnumbers}}}%
5450   {%
5451     \ifcsundef{#1groupname}{\def#2{#1}}{\letcs#2{#1groupname}}%
5452   }%
5453   {%
5454     \def#2{#1}%
5455   }%
5456 }%
5457 }

```

`@getothergrouptitle` Version for the no-indexing app option:

```

5458 \newcommand*{\@gls@noidx@getgrouptitle}[2]{%
5459   \DTLifint{#1}%
5460   {\edef#2{\char#1\relax}}%
5461   {%
5462     \ifcsundef{#1groupname}{\def#2{#1}}{\letcs#2{#1groupname}}%
5463   }%
5464 }

```

`\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`

```

5465 \newcommand*{\glsgetgrouplabel}[1]{%
5466 \ifthenelse{\equal{#1}{\glssymbolsgroupname}}{\glssymbols}{%
5467 \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`

```

5468 \newcommand*{\setentrycounter}[2][ ]{%
5469   \def\@glo@counterprefix{#1}%
5470   \ifx\@glo@counterprefix\@empty
5471     \def\@glo@counterprefix{.}%
5472   \else

```

```

5473 \def\@glo@counterprefix{. #1.}%
5474 \fi
5475 \def\glstentrycounter{#2}%
5476 }

```

The current glossary style can be set using `\setglossarystyle{<style>}`.

`\setglossarystyle`

```

5477 \newcommand*{\setglossarystyle}[1]{%
5478 \ifcsundef{@glsstyle@#1}%
5479 {%
5480 \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
5481 }%
5482 {%
5483 \csname @glsstyle@#1\endcsname
5484 }%
5485 }

```

`\glossarystyle`

```

5486 \newcommand*{\glossarystyle}[1]{%
5487 \ifcsundef{@glsstyle@#1}%
5488 {%
5489 \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
5490 }%
5491 {%
5492 \GlossariesWarning
5493 {Deprecated command \string\glossarystyle.^^J
5494 I recommend you switch to \string\setglossarystyle\space unless
5495 you want to maintain backward compatibility}%
5496 \setglossentrycompatibility
5497 \csname @glsstyle@#1\endcsname

5498 \ifcsdef{@glscompstyle@#1}%
5499 {\setglossentrycompatibility\csuse{@glscompstyle@#1}}%
5500 {}%
5501 }%
5502 }

```

`\newglossarystyle` New glossary styles can be defined using:

`\newglossarystyle{<name>}{<definition>}`

The *<definition>* argument should redefine theglossary, \glossaryheader, \glsgroupheading, \glossaryentryfield and \glsgroupskip (see [subsection 1.18](#) for the definitions of predefined styles). Glossary styles should not redefine \glossarypreamble and \glossarypostamble, as the user should be able to switch between styles without affecting the pre- and postambles.

```

5503 \newcommand{\newglossarystyle}[2]{%
5504 \ifcsundef{@glsstyle@#1}%

```

```

5505  {%
5506    \expandafter\def\csname @glsstyle@#1\endcsname{#2}%
5507  }%
5508  {%
5509    \PackageError{glossaries}{Glossary style ‘#1’ is already defined}{}%
5510  }%
5511 }

```

`\renewglossarystyle` Code for this macro supplied by Marco Daniel.

```

5512 \newcommand{\renewglossarystyle}[2]{%
5513   \ifcsundef{@glsstyle@#1}%
5514   {%
5515     \PackageError{glossaries}{Glossary style ‘#1’ isn’t already defined}{}%
5516   }%
5517   {%
5518     \csdef{@glsstyle@#1}{#2}%
5519   }%
5520 }

```

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`

```

5521 \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`

```

5522 \ifcsundef{hyperlink}%
5523 {%
5524   \def\glshypernumber#1{#1}%

```

```

5525 }%
5526 {%
5527   \def\glshypernumber#1{\@glshypernumber#1\nohyperpage{}}\@nil}
5528 }

```

`\@glshypernumber` This code was provided by Heiko Oberdiek to allow material to be attached to the location.

```

5529 \def\@glshypernumber#1\nohyperpage#2#3\@nil{%
5530   \ifx\#1\%
5531     \else
5532       \@delimR#1\delimR\delimR\%
5533     \fi
5534   \ifx\#2\%
5535     \else
5536       #2%
5537     \fi
5538   \ifx\#3\%
5539     \else
5540       \@glshypernumber#3\@nil
5541     \fi
5542 }

```

`\@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`

```

5543 \def\@delimR#1\delimR #2\delimR #3\{%
5544 \ifx\#2\%
5545   \@delimN{#1}%
5546 \else
5547   \@gls@numberlink{#1}\delimR\@gls@numberlink{#2}%
5548 \fi}

```

`\@delimN` displays a list of individual numbers, instead of a range:

`\@delimN`

```

5549 \def\@delimN#1{\@delimN#1\delimN \delimN\%
5550 \def\@delimN#1\delimN #2\delimN#3\{%
5551 \ifx\#3\%
5552   \@gls@numberlink{#1}%
5553 \else
5554   \@gls@numberlink{#1}\delimN\@gls@numberlink{#2}%
5555 \fi
5556 }

```

The following code is modified from hyperref's `\HyInd@pagelink` where the name of the counter being used is given by `\@gls@counter`.

```

5557 \def\@gls@numberlink#1{%
5558 \begingroup

```

```

5559 \toks@={}%
5560 \@gls@removespaces#1 \@nil
5561 \endgroup}

5562 \def\@gls@removespaces#1 #2\@nil{%
5563 \toks@=\expandafter{\the\toks@#1}%
5564 \ifx\#2\%
5565 \edef\x{\the\toks@}%
5566 \ifx\x\empty
5567 \else

5568 \hyperlink{\glstrycounter\@glo@counterprefix\the\toks@}%
5569 {\the\toks@}%
5570 \fi
5571 \else
5572 \@gls@ReturnAfterFi{%
5573 \@gls@removespaces#2\@nil
5574 }%
5575 \fi
5576 }
5577 \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`

```
5578 \newcommand*\hyperrm[1]{\textrm{\glshypernumber{#1}}}
```

`\hypersf`

```
5579 \newcommand*\hypersf[1]{\textsf{\glshypernumber{#1}}}
```

`\hypertt`

```
5580 \newcommand*\hypertt[1]{\texttt{\glshypernumber{#1}}}
```

`\hyperbf`

```
5581 \newcommand*\hyperbf[1]{\textbf{\glshypernumber{#1}}}
```

`\hypermd`

```
5582 \newcommand*\hypermd[1]{\textmd{\glshypernumber{#1}}}
```

`\hyperit`

```
5583 \newcommand*\hyperit[1]{\textit{\glshypernumber{#1}}}
```

`\hypersl`

```
5584 \newcommand*\hypersl[1]{\textsl{\glshypernumber{#1}}}
```

`\hyperup`

```
5585 \newcommand*\hyperup[1]{\textup{\glshypernumber{#1}}}
```

`\hypersc`

```
5586 \newcommand*{\hypersc}[1]{\textsc{\glshypernumber{#1}}}
```

`\hyperemph`

```
5587 \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.

```
5588 \newcommand{\oldacronym}[4][\gls@label]{%
5589   \def\gls@label{#2}%
5590   \newacronym[#4]{#1}{#2}{#3}%
5591   \ifcsundef{xspace}%
5592     {%
5593       \expandafter\edef\csname#1\endcsname{%
5594         \noexpand\@ifstar{\noexpand\Gls{#1}}{\noexpand\gls{#1}}}%
5595     }%
5596   }%
5597   {%
5598     \expandafter\edef\csname#1\endcsname{%
5599       \noexpand\@ifstar{\noexpand\Gls{#1}\noexpand\xspace}{%
5600         \noexpand\gls{#1}\noexpand\xspace}%
5601     }%
5602   }%
5603 }
```

```
\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`

```
5604 \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.

```
5605 \newcommand*{\acrpluralsuffix}{\glspluralsuffix}
```

If `garamondx` has been loaded, need to use `\textulc` instead of `\textup`.

`\glstextup`

```
5606 \newrobustcmd*{\glstextup}[1]{\ifdef\textulc{\textulc{#1}}{\textup{#1}}}
```

The following are defined for compatibility with version 2.07 and earlier.

`\glsshortkey`

```
5607 \newcommand*{\glsshortkey}{short}
```

`\glsshortpluralkey`

```
5608 \newcommand*{\glsshortpluralkey}{shortplural}
```

`\glslongkey`

```
5609 \newcommand*{\glslongkey}{long}
```

`\glslongpluralkey`

```
5610 \newcommand*{\glslongpluralkey}{longplural}
```

`\acrfull` Full form of the acronym.

```
5611 \newrobustcmd*{\acrfull}{%
```

```
5612 \ifstar\s@acrfull\ns@acrfull
```

```
5613 }
```



```

5614 \newcommand*\s@acrfull[2][{}]{%
5615   \new@ifnextchar[{\@acrfull{hyper=false,#1}{#2}}}%
5616   {\@acrfull{hyper=false,#1}{#2}[]}%
5617 }
5618 \newcommand*\ns@acrfull[2][{}]{%
5619   \new@ifnextchar[{\@acrfull{#1}{#2}}}%
5620   {\@acrfull{#1}{#2}[]}%
5621 }

```

`\@acrfull` Low-level macro:

```

5622 \def\@acrfull#1#2[#3]{%
    Make it easier for acronym styles to change this:
5623   \acrfullfmt{#1}{#2}{#3}%
5624 }

```

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.

```

5625 \newcommand*\@acrfullfmt[3]{%
5626   \acrlinkfullformat{\@acrlong}{\@acrshort}{#1}{#2}{#3}%
5627 }

```

`\acrlinkfullformat` Format for full links like `\acrfull`. Syntax: `\acrlinkfullformat{<long cs>}{<short cs>}{<options>}{<label>}{<insert>}`

```

5628 \newcommand\@acrlinkfullformat[5]{%
5629   \acrfullformat{#1}{#3}{#4}{#5}{#2}{#3}{#4}[]}%
5630 }

```

`\acrfullformat` Default full form is `<long>` (`<short>`).

```

5631 \newcommand\@acrfullformat[2]{#1\space(#2)}

```

Default format for full acronym

`\Acrfull`

```

5632 \newrobustcmd*\Acrfull{}%
5633   \@ifstar\s@Acrfull\ns@Acrfull
5634 }

5635 \newcommand*\s@Acrfull[2][{}]{%
5636   \new@ifnextchar[{\@Acrfull{hyper=false,#1}{#2}}}%
5637   {\@Acrfull{hyper=false,#1}{#2}[]}%
5638 }
5639 \newcommand*\ns@Acrfull[2][{}]{%
5640   \new@ifnextchar[{\@Acrfull{#1}{#2}}}%
5641   {\@Acrfull{#1}{#2}[]}%
5642 }

```

Low-level macro:

```
5643 \def\@Acrfull#1#2[#3]{%
```

Make it easier for acronym styles to change this:

```
5644 \Acrfullfmt{#1}{#2}{#3}%  
5645 }
```

`\Acrfullfmt` First letter upper case full format.

```
5646 \newcommand*\@Acrfullfmt}[3]{%  
5647 \acrlinkfullformat{\@Acrlong}{\@acrshort}{#1}{#2}{#3}%  
5648 }
```

`\ACRfull`

```
5649 \newrobustcmd*\@ACRfull}{%  
5650 \@ifstar\s@ACRfull\@ns@ACRfull  
5651 }  
  
5652 \newcommand*\s@ACRfull[2][]{%  
5653 \new@ifnextchar[{\@ACRfull{hyper=false,#1}{#2}}%  
5654 {\@ACRfull{hyper=false,#1}{#2}[]}%  
5655 }  
5656 \newcommand*\ns@ACRfull[2][]{%  
5657 \new@ifnextchar[{\@ACRfull{#1}{#2}}%  
5658 {\@ACRfull{#1}{#2}[]}%  
5659 }
```

Low-level macro:

```
5660 \def\@ACRfull#1#2[#3]{%
```

Make it easier for acronym styles to change this:

```
5661 \ACRfullfmt{#1}{#2}{#3}%  
5662 }
```

`\ACRfullfmt` All upper case full format.

```
5663 \newcommand*\@ACRfullfmt}[3]{%  
5664 \acrlinkfullformat{\@ACRlong}{\@ACRshort}{#1}{#2}{#3}%  
5665 }
```

Plural:

`\acrfullpl`

```
5666 \newrobustcmd*\@acrfullpl}{%  
5667 \@ifstar\s@acrfullpl\@ns@acrfullpl  
5668 }  
  
5669 \newcommand*\s@acrfullpl[2][]{%  
5670 \new@ifnextchar[{\@acrfullpl{hyper=false,#1}{#2}}%  
5671 {\@acrfullpl{hyper=false,#1}{#2}[]}%  
5672 }  
5673 \newcommand*\ns@acrfullpl[2][]{%
```

```

5674 \new@ifnextchar[{\@acrfullpl{#1}{#2}}%
5675         {\@acrfullpl{#1}{#2} []}%
5676 }

```

Low-level macro:

```

5677 \def\@acrfullpl#1#2[#3]{%

```

Make it easier for acronym styles to change this:

```

5678 \acrfullplfmt{#1}{#2}{#3}%
5679 }

```

`\acrfullplfmt` No case change plural full format.

```

5680 \newcommand*{\acrfullplfmt}[3]{%
5681 \acrlinkfullformat{\@acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%
5682 }

```

`\Acrfullpl`

```

5683 \newrobustcmd*{\Acrfullpl}{%
5684 \ifstar\s@Acrfullpl\ns@Acrfullpl
5685 }

5686 \newcommand*\s@Acrfullpl[2][]{%
5687 \new@ifnextchar[{\@Acrfullpl{hyper=false,#1}{#2}}%
5688         {\@Acrfullpl{hyper=false,#1}{#2} []}%
5689 }
5690 \newcommand*\ns@Acrfullpl[2][]{%
5691 \new@ifnextchar[{\@Acrfullpl{#1}{#2}}%
5692         {\@Acrfullpl{#1}{#2} []}%
5693 }

```

Low-level macro:

```

5694 \def\@Acrfullpl#1#2[#3]{%

```

Make it easier for acronym styles to change this:

```

5695 \Acrfullplfmt{#1}{#2}{#3}%
5696 }

```

`\Acrfullplfmt` First letter upper case plural full format.

```

5697 \newcommand*{\Acrfullplfmt}[3]{%
5698 \acrlinkfullformat{\@acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%
5699 }

```

`\ACRfullpl`

```

5700 \newrobustcmd*{\ACRfullpl}{%
5701 \ifstar\s@ACRfullpl\ns@ACRfullpl
5702 }

5703 \newcommand*\s@ACRfullpl[2][]{%
5704 \new@ifnextchar[{\@ACRfullpl{hyper=false,#1}{#2}}%
5705         {\@ACRfullpl{hyper=false,#1}{#2} []}%

```

```

5706 }
5707 \newcommand*\ns@ACRfullpl[2] [] {%
5708   \new@ifnextchar[{\@ACRfullpl{#1}{#2}}%
5709     {\@ACRfullpl{#1}{#2} []}%
5710 }

```

Low-level macro:

```

5711 \def\@ACRfullpl#1#2[#3] {%
    Make it easier for acronym styles to change this:
5712   \ACRfullplfmt{#1}{#2}{#3}%
5713 }

```

`\ACRfullplfmt` All upper case plural full format.

```

5714 \newcommand*\@ACRfullplfmt[3] {%
5715   \acrlinkfullformat{\@ACRlongpl}{\@ACRshortpl}{#1}{#2}{#3}%
5716 }

```

## 1.17 Predefined acronym styles

`\acronymfont` This is only used with the additional acronym styles:

```

5717 \newcommand{\acronymfont}[1]{#1}

```

`\firstacronymfont` This is only used with the additional acronym styles:

```

5718 \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.

```

5719 \newcommand*\acrnameformat[2]{\acronymfont{#1}}

```

Define some tokens used by `\newacronym`:

`\glskeylisttok`

```

5720 \newtoks\glskeylisttok

```

`\glslabeltok`

```

5721 \newtoks\glslabeltok

```

`\glsshorttok`

```

5722 \newtoks\glsshorttok

```

`\glslongtok`

```

5723 \newtoks\glslongtok

```

`\newacronymhook` Provide a hook for `\newacronym`:

```

5724 \newcommand*\newacronymhook{}

```

etGenericNewAcronym New improved version of setting the acronym style.

```

5725 \newcommand*\SetGenericNewAcronym{%
5726   \renewcommand{\newacronym}[4][\{%
5727     \ifdefempty{\@glsacronymlists}%
5728     {%
5729       \def\@glo@type{\acronymtype}%
5730       \setkeys{glossentry}{##1}%
5731       \DeclareAcronymList{\@glo@type}%
5732     }%
5733   }%
5734   \glskeylisttok{##1}%
5735   \glslabeltok{##2}%
5736   \glsshorttok{##3}%
5737   \glslongtok{##4}%
5738   \newacronymhook
5739   \protected@edef\@do@newglossaryentry{%
5740     \noexpand\newglossaryentry{\the\glslabeltok}%
5741     {%
5742       type=\acronymtype,%
5743       name={\expandonce{\acronymentry{##2}}},%
5744       sort={\acronymsort{\the\glsshorttok}{\the\glslongtok}},%
5745       text={\the\glsshorttok},%
5746       short={\the\glsshorttok},%
5747       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5748       long={\the\glslongtok},%
5749       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5750       \GenericAcronymFields,%
5751       \the\glskeylisttok
5752     }%
5753   }%
5754   \@do@newglossaryentry
5755 }%

```

Make sure that \acrfull etc reflects the new style:

```

5756 \renewcommand*\acrfullfmt}[3]{%
5757   \glslink{##1}{##2}{\genacrfullformat{##2}{##3}}}%
5758 \renewcommand*\Acrfullfmt}[3]{%
5759   \glslink{##1}{##2}{\Genacrfullformat{##2}{##3}}}%
5760 \renewcommand*\ACRfullfmt}[3]{%
5761   \glslink{##1}{##2}{%
5762     \mfirstucMakeUppercase{\genacrfullformat{##2}{##3}}}%
5763 \renewcommand*\acrfullplfmt}[3]{%
5764   \glslink{##1}{##2}{\genplacrfullformat{##2}{##3}}}%
5765 \renewcommand*\Acrfullplfmt}[3]{%
5766   \glslink{##1}{##2}{\Genplacrfullformat{##2}{##3}}}%
5767 \renewcommand*\ACRfullplfmt}[3]{%
5768   \glslink{##1}{##2}{%
5769     \mfirstucMakeUppercase{\genplacrfullformat{##2}{##3}}}%

```

Make sure that \glsentryfull etc reflects the new style:

```

5770 \renewcommand*{\glsentryfull}[1]{\genacrfullformat{##1}{}}%
5771 \renewcommand*{\Glsentryfull}[1]{\Genacrfullformat{##1}{}}%
5772 \renewcommand*{\glsentryfullpl}[1]{\genplacrfullformat{##1}{}}%
5773 \renewcommand*{\Glsentryfullpl}[1]{\Genplacrfullformat{##1}{}}%
5774 }

```

`\GenericAcronymFields` Fields used by `\SetGenericNewAcronym` that can be changed by the acronym style.

```

5775 \newcommand*{\GenericAcronymFields}{description={\the\glslongtok}}

```

`\acronymentry` `\acronymentry{<label>}`

Display style for the name field in the list of acronyms.

```

5776 \newcommand*{\acronymentry}[1]{\acronymfont{\glsentryshort{#1}}}

```

`\acronymsort` `\acronymsort{<short>}{<long>}`

Default sort format for acronyms.

```

5777 \newcommand*{\acronymsort}[2]{#1}

```

`\setacronymstyle` `\setacronymstyle{<style name>}`

```

5778 \newcommand*{\setacronymstyle}[1]{%
5779   \ifcsundef{@glsacr@dispstyle@#1}%
5780   {%
5781     \PackageError{glossaries}{Undefined acronym style ‘#1’}{%
5782     }%
5783   {%
5784     \ifdefempty{\@glsacronymlists}%
5785     {%
5786       \DeclareAcronymList{\acronymtype}%
5787     }%
5788     {%
5789       \SetGenericNewAcronym
5790       \GlsUseAcrStyleDefs{#1}%
5791       \@for\@gls@type:=\@glsacronymlists\do{%
5792         \defglsentryfmt[\@gls@type]{\GlsUseAcrEntryDispStyle{#1}}%
5793       }%
5794     }%
5795 }

```

`\newacronymstyle` `\newacronymstyle{<style name>}{<entry format definition>}{<display definitions>}`

Defines a new acronym style called *<style name>*.

```

5796 \newcommand*{\newacronymstyle}[3]{%
5797   \ifcsdef{@glsacr@dispstyle@#1}%
5798   {%
5799     \PackageError{glossaries}{Acronym style ‘#1’ already exists}{}%
5800   }%
5801   {%
5802     \csdef{@glsacr@dispstyle@#1}{#2}%
5803     \csdef{@glsacr@styledefs@#1}{#3}%
5804   }%
5805 }
```

`\renewacronymstyle` Redefines the given acronym style.

```

5806 \newcommand*{\renewacronymstyle}[3]{%
5807   \ifcsdef{@glsacr@dispstyle@#1}%
5808   {%
5809     \csdef{@glsacr@dispstyle@#1}{#2}%
5810     \csdef{@glsacr@styledefs@#1}{#3}%
5811   }%
5812   {%
5813     \PackageError{glossaries}{Acronym style ‘#1’ doesn’t exist}{}%
5814   }%
5815 }
```

`\useAcrEntryDispStyle`

```

5816 \newcommand*{\GlsUseAcrEntryDispStyle}[1]{\csuse{@glsacr@dispstyle@#1}}
```

`\GlsUseAcrStyleDefs`

```

5817 \newcommand*{\GlsUseAcrStyleDefs}[1]{\csuse{@glsacr@styledefs@#1}}
```

Predefined acronym styles:

`long-short` *<long>* (*<short>*) acronym style.

```

5818 \newacronymstyle{long-short}%
5819 {%
```

Check for long form in case this is a mixed glossary.

```

5820   \ifglshaslong{\glslabel}{\glsngenacfmt}{\glsngenentryfmt}%
5821 }%
5822 {%
5823   \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
5824   \renewcommand*{\genacrfullformat}[2]{%
5825     \glsentrylong{##1}##2\space
5826     (\protect\firstacronymfont{\glsentryshort{##1}})%
5827   }%
5828   \renewcommand*{\Genacrfullformat}[2]{%
5829     \Glsentrylong{##1}##2\space
5830     (\protect\firstacronymfont{\glsentryshort{##1}})%
5831   }%
```

```

5832 \renewcommand*{\genplacrfullformat}[2]{%
5833   \glentrylongpl{##1}##2\space
5834   (\protect\firstacronymfont{\glentryshortpl{##1}})%
5835 }%
5836 \renewcommand*{\Genplacrfullformat}[2]{%
5837   \Glentrylongpl{##1}##2\space
5838   (\protect\firstacronymfont{\glentryshortpl{##1}})%
5839 }%
5840 \renewcommand*{\acronymentry}[1]{\acronymfont{\glentryshort{##1}}}
5841 \renewcommand*{\acronymsort}[2]{##1}%
5842 \renewcommand*{\acronymfont}[1]{##1}%
5843 \renewcommand*{\firstacronymfont}[1]{\acronymfont{##1}}%
5844 \renewcommand*{\acrpluralsuffix}{\glpluralsuffix}%
5845 }

```

short-long <short> (<long>) acronym style.

```

5846 \newacronymstyle{short-long}%
5847 {%
5848   Check for long form in case this is a mixed glossary.
5849   \ifglshaslong{\glslabel}{\glsgenacfmt}{\glsgenentryfmt}%
5850 }%
5851 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
5852 \renewcommand*{\genacrfullformat}[2]{%
5853   \protect\firstacronymfont{\glentryshort{##1}}##2\space
5854   (\glentrylong{##1})%
5855 }%
5856 \renewcommand*{\Genacrfullformat}[2]{%
5857   \protect\firstacronymfont{\Glentryshort{##1}}##2\space
5858   (\glentrylong{##1})%
5859 }%
5860 \renewcommand*{\genplacrfullformat}[2]{%
5861   \protect\firstacronymfont{\glentryshortpl{##1}}##2\space
5862   (\glentrylongpl{##1})%
5863 }%
5864 \renewcommand*{\Genplacrfullformat}[2]{%
5865   \protect\firstacronymfont{\Glentryshortpl{##1}}##2\space
5866   (\glentrylongpl{##1})%
5867 }%
5868 \renewcommand*{\acronymentry}[1]{\acronymfont{\glentryshort{##1}}}
5869 \renewcommand*{\acronymsort}[2]{##1}%
5870 \renewcommand*{\acronymfont}[1]{##1}%
5871 \renewcommand*{\firstacronymfont}[1]{\acronymfont{##1}}%
5872 \renewcommand*{\acrpluralsuffix}{\glpluralsuffix}%
5873 }

```

long-sc-short <long> (\textsc{<short>}) acronym style.

```

5874 \newacronymstyle{long-sc-short}%

```



```

5875 {%
5876   \GlsUseAcrEntryDispStyle{long-short}%
5877 }%
5878 {%
5879   \GlsUseAcrStyleDefs{long-short}%
5880   \renewcommand{\acronymfont}[1]{\textsc{##1}}%
5881   \renewcommand*{\acrpluralsuffix}{\glstextup{\glspluralsuffix}}%
5882 }

```

long-sm-short    *<long>* (\textsmaller{<short>}) acronym style.

```

5883 \newacronymstyle{long-sm-short}%
5884 {%
5885   \GlsUseAcrEntryDispStyle{long-short}%
5886 }%
5887 {%
5888   \GlsUseAcrStyleDefs{long-short}%
5889   \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
5890   \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
5891 }

```

sc-short-long    *<short>* (\textsc{<long>}) acronym style.

```

5892 \newacronymstyle{sc-short-long}%
5893 {%
5894   \GlsUseAcrEntryDispStyle{short-long}%
5895 }%
5896 {%
5897   \GlsUseAcrStyleDefs{short-long}%
5898   \renewcommand{\acronymfont}[1]{\textsc{##1}}%
5899   \renewcommand*{\acrpluralsuffix}{\glstextup{\glspluralsuffix}}%
5900 }

```

sm-short-long    *<short>* (\textsmaller{<long>}) acronym style.

```

5901 \newacronymstyle{sm-short-long}%
5902 {%
5903   \GlsUseAcrEntryDispStyle{short-long}%
5904 }%
5905 {%
5906   \GlsUseAcrStyleDefs{short-long}%
5907   \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
5908   \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
5909 }

```

long-short-desc    *<long>* ({<short>}) acronym style that has an accompanying description (which the user needs to supply).

```

5910 \newacronymstyle{long-short-desc}%
5911 {%
5912   \GlsUseAcrEntryDispStyle{long-short}%
5913 }%
5914 {%

```

```

5915 \GlsUseAcrStyleDefs{long-short}%
5916 \renewcommand*{\GenericAcronymFields}{}%
5917 \renewcommand*{\acronymsort}[2]{##2}%
5918 \renewcommand*{\acronymentry}[1]{%
5919     \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
5920 }

```

long-sc-short-desc    *<long>* (\textsc{<short>}) acronym style that has an accompanying description (which the user needs to supply).

```

5921 \newacronymstyle{long-sc-short-desc}%
5922 {%
5923     \GlsUseAcrEntryDisplayStyle{long-sc-short}%
5924 }%
5925 {%
5926     \GlsUseAcrStyleDefs{long-sc-short}%
5927     \renewcommand*{\GenericAcronymFields}{}%
5928     \renewcommand*{\acronymsort}[2]{##2}%
5929     \renewcommand*{\acronymentry}[1]{%
5930         \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
5931 }

```

long-sm-short-desc    *<long>* (\textsmaller{<short>}) acronym style that has an accompanying description (which the user needs to supply).

```

5932 \newacronymstyle{long-sm-short-desc}%
5933 {%
5934     \GlsUseAcrEntryDisplayStyle{long-sm-short}%
5935 }%
5936 {%
5937     \GlsUseAcrStyleDefs{long-sm-short}%
5938     \renewcommand*{\GenericAcronymFields}{}%
5939     \renewcommand*{\acronymsort}[2]{##2}%
5940     \renewcommand*{\acronymentry}[1]{%
5941         \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
5942 }

```

short-long-desc    *<short>* ({<long>}) acronym style that has an accompanying description (which the user needs to supply).

```

5943 \newacronymstyle{short-long-desc}%
5944 {%
5945     \GlsUseAcrEntryDisplayStyle{short-long}%
5946 }%
5947 {%
5948     \GlsUseAcrStyleDefs{short-long}%
5949     \renewcommand*{\GenericAcronymFields}{}%
5950     \renewcommand*{\acronymsort}[2]{##2}%
5951     \renewcommand*{\acronymentry}[1]{%
5952         \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
5953 }

```

sc-short-long-desc    *<long>* (\textsc{<short>}) acronym style that has an accompanying description (which the user needs to supply).

```

5954 \newacronymstyle{sc-short-long-desc}%
5955 {%
5956   \GlsUseAcrEntryDispStyle{sc-short-long}%
5957 }%
5958 {%
5959   \GlsUseAcrStyleDefs{sc-short-long}%
5960   \renewcommand*{\GenericAcronymFields}{}%
5961   \renewcommand*{\acronymsort}[2]{##2}%
5962   \renewcommand*{\acronymentry}[1]{%
5963     \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
5964 }

```

sm-short-long-desc    *<long>* (\textsmaller{<short>}) acronym style that has an accompanying description (which the user needs to supply).

```

5965 \newacronymstyle{sm-short-long-desc}%
5966 {%
5967   \GlsUseAcrEntryDispStyle{sm-short-long}%
5968 }%
5969 {%
5970   \GlsUseAcrStyleDefs{sm-short-long}%
5971   \renewcommand*{\GenericAcronymFields}{}%
5972   \renewcommand*{\acronymsort}[2]{##2}%
5973   \renewcommand*{\acronymentry}[1]{%
5974     \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
5975 }

```

dua    *<long>* only acronym style.

```

5976 \newacronymstyle{dua}%
5977 {%

```

    Check for long form in case this is a mixed glossary.

```

5978   \ifdefempty\glscustomtext
5979   {%
5980     \ifglshaslong{\glslabel}%
5981     {%
5982       \glcifplural
5983       {%

```

        Plural form:

```

5984         \glscapscase
5985         {%

```

        Plural form, don't adjust case:

```

5986         \glentrylongpl{\glslabel}\glinsert
5987         }%
5988         {%

```

        Plural form, make first letter upper case:

```

5989         \Glsentrylongpl{\glslabel}\glsinsert
5990     }%
5991     {%

Plural form, all caps:
5992         \mfirstucMakeUppercase
5993         {\glsentrylongpl{\glslabel}\glsinsert}%
5994     }%
5995 }%
5996 {%

Singular form
5997     \glscapscase
5998     {%

Singular form, don't adjust case:
5999         \glsentrylong{\glslabel}\glsinsert
6000     }%
6001     {%

Subsequent singular form, make first letter upper case:
6002         \Glsentrylong{\glslabel}\glsinsert
6003     }%
6004     {%

Subsequent singular form, all caps:
6005         \mfirstucMakeUppercase
6006         {\glsentrylong{\glslabel}\glsinsert}%
6007     }%
6008 }%
6009 }%
6010 {%

Not an acronym:
6011     \glsgenentryfmt
6012 }%
6013 }%
6014 {\glscustomtext\glsinsert}%
6015}%
6016{%
6017 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%

6018 \renewcommand*{\acrfullfmt}[3]{%
6019     \glslink[##1]{##2}{\glsentrylong{##2}##3\space
6020     (\acronymfont{\glsentryshort{##2}})}}%
6021 \renewcommand*{\Acrfullfmt}[3]{%
6022     \glslink[##1]{##2}{\Glsentrylong{##2}##3\space
6023     (\acronymfont{\glsentryshort{##2}})}}%
6024 \renewcommand*{\ACRfullfmt}[3]{%
6025     \glslink[##1]{##2}{%
6026     \mfirstucMakeUppercase{\glsentrylong{##2}##3\space
6027     (\acronymfont{\glsentryshort{##2}})}}}%

```

```

6028 \renewcommand*{\acrfullplfmt}[3]{%
6029   \glslink{##1}{##2}{\glsentrylongpl{##2}##3\space
6030     (\acronymfont{\glsentryshortpl{##2}})}}%

6031 \renewcommand*{\Acrfullplfmt}[3]{%
6032   \glslink{##1}{##2}{\Glsentrylongpl{##2}##3\space
6033     (\acronymfont{\glsentryshortpl{##2}})}}%
6034 \renewcommand*{\ACRfullplfmt}[3]{%
6035   \glslink{##1}{##2}{%
6036     \mfirstucMakeUppercase{\glsentrylongpl{##2}##3\space
6037       (\acronymfont{\glsentryshortpl{##2}})}}}%
6038 \renewcommand*{\glsentryfull}[1]{%
6039   \glsentrylong{##1}\space(\acronymfont{\glsentryshort{##1}})%
6040 }%
6041 \renewcommand*{\Glsentryfull}[1]{%
6042   \Glsentrylong{##1}\space(\acronymfont{\glsentryshort{##1}})%
6043 }%
6044 \renewcommand*{\glsentryfullpl}[1]{%
6045   \glsentrylongpl{##1}\space(\acronymfont{\glsentryshortpl{##1}})%
6046 }%
6047 \renewcommand*{\Glsentryfullpl}[1]{%
6048   \Glsentrylongpl{##1}\space(\acronymfont{\glsentryshortpl{##1}})%
6049 }%
6050 \renewcommand*{\acronymentry}[1]{\acronymfont{\glsentryshort{##1}}}%
6051 \renewcommand*{\acronymsort}[2]{##1}%
6052 \renewcommand*{\acronymfont}[1]{##1}%
6053 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
6054 }

```

dua-desc *<long>* only acronym style with user-supplied description.

```

6055 \newacronymstyle{dua-desc}%
6056 {%
6057   \GlsUseAcrEntryDisplayStyle{dua}%
6058 }%
6059 {%
6060   \GlsUseAcrStyleDefs{dua}%
6061   \renewcommand*{\GenericAcronymFields}{}%

6062   \renewcommand*{\acronymentry}[1]{\acronymfont{\glsentrylong{##1}}}%
6063   \renewcommand*{\acronymsort}[2]{##2}%
6064 }%

```

footnote *<short>*\footnote{*<long>*} acronym style.

```

6065 \newacronymstyle{footnote}%
6066 {%

  Check for long form in case this is a mixed glossary.

6067   \ifglshaslong{\glslabel}{\glsngenacfmt}{\glsngenentryfmt}%
6068 }%
6069 {%
6070   \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%

```

Need to ensure hyperlinks are switched off on first use:

```

6071 \glshyperfirstfalse
6072 \renewcommand*{\genacrfullformat}[2]{%
6073   \protect\firstacronymfont{\glsentryshort{##1}}##2%
6074   \protect\footnote{\glsentrylong{##1}}%
6075 }%
6076 \renewcommand*{\Genacrfullformat}[2]{%
6077   \firstacronymfont{\Glsentryshort{##1}}##2%
6078   \protect\footnote{\glsentrylong{##1}}%
6079 }%
6080 \renewcommand*{\genplacrfullformat}[2]{%
6081   \protect\firstacronymfont{\glsentryshortpl{##1}}##2%
6082   \protect\footnote{\glsentrylongpl{##1}}%
6083 }%
6084 \renewcommand*{\Genplacrfullformat}[2]{%
6085   \protect\firstacronymfont{\Glsentryshortpl{##1}}##2%
6086   \protect\footnote{\glsentrylongpl{##1}}%
6087 }%
6088 \renewcommand*{\acronymentry}[1]{\acronymfont{\glsentryshort{##1}}}%
6089 \renewcommand*{\acronymsort}[2]{##1}%
6090 \renewcommand*{\acronymfont}[1]{##1}%
6091 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%

```

Don't use footnotes for \acrfull:

```

6092 \renewcommand*{\acrfullfmt}[3]{%
6093   \glslink[##1]{##2}{\acronymfont{\glsentryshort{##2}}##3\space
6094   (\glsentrylong{##2})}%
6095 \renewcommand*{\Acrfullfmt}[3]{%
6096   \glslink[##1]{##2}{\acronymfont{\Glsentryshort{##2}}##3\space
6097   (\glsentrylong{##2})}%
6098 \renewcommand*{\ACRfullfmt}[3]{%
6099   \glslink[##1]{##2}{%
6100     \mfirstucMakeUppercase{\acronymfont{\glsentryshort{##2}}##3\space
6101     (\glsentrylong{##2})}}}%
6102 \renewcommand*{\acrfullplfmt}[3]{%
6103   \glslink[##1]{##2}{\acronymfont{\glsentryshortpl{##2}}##3\space
6104   (\glsentrylongpl{##2})}%
6105 \renewcommand*{\Acrfullplfmt}[3]{%
6106   \glslink[##1]{##2}{\acronymfont{\Glsentryshortpl{##2}}##3\space
6107   (\glsentrylongpl{##2})}%
6108 \renewcommand*{\ACRfullplfmt}[3]{%
6109   \glslink[##1]{##2}{%
6110     \mfirstucMakeUppercase{\acronymfont{\glsentryshortpl{##2}}##3\space
6111     (\glsentrylongpl{##2})}}}%

```

Similarly for \glsentryfull etc:

```

6112 \renewcommand*{\glsentryfull}[1]{%
6113   \acronymfont{\glsentryshort{##1}}\space(\glsentrylong{##1})}%
6114 \renewcommand*{\Glsentryfull}[1]{%
6115   \acronymfont{\Glsentryshort{##1}}\space(\glsentrylong{##1})}%

```

```

6116 \renewcommand*{\glsentryfullpl}[1]{%
6117     \acronymfont{\glsentryshortpl{##1}}\space(\glsentrylongpl{##1})}%
6118 \renewcommand*{\Glsentryfullpl}[1]{%
6119     \acronymfont{\Glsentryshortpl{##1}}\space(\glsentrylongpl{##1})}%
6120 }

```

footnote-sc \textsc{<short>}\footnote{<long>} acronym style.

```

6121 \newacronymstyle{footnote-sc}%
6122 {%
6123     \GlsUseAcrEntryDisplayStyle{footnote}%
6124 }%
6125 {%
6126     \GlsUseAcrStyleDefs{footnote}%
6127     \renewcommand{\acronymentry}[1]{\acronymfont{\glsentryshort{##1}}}
6128     \renewcommand{\acronymfont}[1]{\textsc{##1}}%
6129     \renewcommand*{\acrpluralsuffix}{\glstextup{\glspluralsuffix}}%
6130 }%

```

footnote-sm \textsmaller{<short>}\footnote{<long>} acronym style.

```

6131 \newacronymstyle{footnote-sm}%
6132 {%
6133     \GlsUseAcrEntryDisplayStyle{footnote}%
6134 }%
6135 {%
6136     \GlsUseAcrStyleDefs{footnote}%
6137     \renewcommand{\acronymentry}[1]{\acronymfont{\glsentryshort{##1}}}
6138     \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
6139     \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
6140 }%

```

footnote-desc <short>\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```

6141 \newacronymstyle{footnote-desc}%
6142 {%
6143     \GlsUseAcrEntryDisplayStyle{footnote}%
6144 }%
6145 {%
6146     \GlsUseAcrStyleDefs{footnote}%
6147     \renewcommand*{\GenericAcronymFields}{}%
6148     \renewcommand*{\acronymsort}[2]{##2}%
6149     \renewcommand*{\acronymentry}[1]{%
6150         \glsentrylong{##1}\space (\acronymfont{\glsentryshort{##1}})%
6151     }%

```

footnote-sc-desc \textsc{<short>}\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```

6152 \newacronymstyle{footnote-sc-desc}%
6153 {%

```

```

6154 \GlsUseAcrEntryDispStyle{footnote-sc}%
6155 }%
6156 {%
6157 \GlsUseAcrStyleDefs{footnote-sc}%
6158 \renewcommand*{\GenericAcronymFields}{}%
6159 \renewcommand*{\acronymsort}[2]{##2}%
6160 \renewcommand*{\acronymentry}[1]{%
6161     \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
6162 }

```

footnote-sm-desc \textsmaller{<short>}\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```

6163 \newacronymstyle{footnote-sm-desc}%
6164 {%
6165 \GlsUseAcrEntryDispStyle{footnote-sm}%
6166 }%
6167 {%
6168 \GlsUseAcrStyleDefs{footnote-sm}%
6169 \renewcommand*{\GenericAcronymFields}{}%
6170 \renewcommand*{\acronymsort}[2]{##2}%
6171 \renewcommand*{\acronymentry}[1]{%
6172     \glentrylong{##1}\space (\acronymfont{\glentryshort{##1}})}%
6173 }

```

fineAcronymSynonyms

```

6174 \newcommand*{\DefineAcronymSynonyms}{%

```

Short form

\acs

```

6175 \let\acs\acrshort

```

First letter uppercase short form

\Acs

```

6176 \let\Acs\Acrshort

```

Plural short form

\acsp

```

6177 \let\acsp\acrshortpl

```

First letter uppercase plural short form

\Acsp

```

6178 \let\Acsp\Acrshortpl

```

Long form

\acl

```

6179 \let\acl\aclong

```



Plural long form

\aclp

6180 \let\aclp\acrlongpl

First letter upper case long form

\Acl

6181 \let\Acl\Acrlong

First letter upper case plural long form

\Ac1p

6182 \let\Ac1p\Acrlongpl

Full form

\acf

6183 \let\acf\acrfull

Plural full form

\acfp

6184 \let\acfp\acrfullpl

First letter upper case full form

\Acf

6185 \let\Acf\Acrfull

First letter upper case plural full form

\Acfp

6186 \let\Acfp\Acrfullpl

Standard form

\ac

6187 \let\ac\gls

First upper case standard form

\Ac

6188 \let\Ac\Gls

Standard plural form

\acp

6189 \let\acp\glspl

Standard first letter upper case plural form

\Acp

6190 \let\Acp\Glspl

6191 }

Define synonyms if required

6192 \ifglsacrshortcuts

6193 \DefineAcronymSynonyms

6194 \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.

6195 \newcommand\*\SetDefaultAcronymDisplayStyle[1]{%

6196 \defglsentryfmt[#1]{\glsentryfmt}%

6197 }

**DefaultNewAcronymDef** Sets up the acronym definition for the default style. The information is provided by the tokens \glslabeltok, \glsshorttok, \glslongtok and \glskeylisttok.

6198 \newcommand\*\DefaultNewAcronymDef{%

6199 \edef\@do@newglossaryentry{%

6200 \noexpand\newglossaryentry{\the\glslabeltok}%

6201 {%

6202 type=\acronymtype,%

6203 name={\the\glsshorttok},%

6204 sort={\the\glsshorttok},%

6205 text={\the\glsshorttok},%

6206 first={\acrfullformat{\the\glslongtok}{\the\glsshorttok}},%

6207 plural={\noexpand\expandonce\noexpand\@glo@shortpl},%

6208 firstplural={\acrfullformat{\noexpand\expandonce\noexpand\@glo@longpl}%

6209 {\noexpand\expandonce\noexpand\@glo@shortpl}},%

6210 short={\the\glsshorttok},%

6211 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%

6212 long={\the\glslongtok},%

6213 longplural={\the\glslongtok\noexpand\acrpluralsuffix},%

6214 description={\the\glslongtok},%

6215 descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%

Remaining options specified by the user:

6216 \the\glskeylisttok

6217 }%

6218 }%

6219 \let\@org@gls@assign@firstpl\gls@assign@firstpl

6220 \let\@org@gls@assign@plural\gls@assign@plural

6221 \let\@org@gls@assign@descplural\gls@assign@descplural

6222 \def\gls@assign@firstpl##1##2{%

6223 \@@gls@expand@field{##1}{firstpl}{##2}%

6224 }%

6225 \def\gls@assign@plural##1##2{%

6226 \@@gls@expand@field{##1}{plural}{##2}%

6227 }%

```

6228 \def\gls@assign@descplural##1##2{%
6229   \@@gls@expand@field{##1}{descplural}{##2}%
6230 }%
6231 \do@newglossaryentry
6232 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6233 \let\gls@assign@plural\@org@gls@assign@plural
6234 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6235 }

```

**DefaultAcronymStyle** Set up the default acronym style:

```

6236 \newcommand*\SetDefaultAcronymStyle{%

```

Set the display style:

```

6237 \@for\@gls@type:=\@gls@acronymlists\do{%
6238   \SetDefaultAcronymDisplayStyle{\@gls@type}%
6239 }%

```

Set up the definition of \newacronym:

```

6240 \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).

```

6241   \ifx\@gls@acronymlists\@empty
6242     \def\@glo@type{\acronymtype}%
6243     \setkeys{glossentry}{##1}%
6244     \DeclareAcronymList{\@glo@type}%
6245     \SetDefaultAcronymDisplayStyle{\@glo@type}%
6246   \fi
6247   \glskeylisttok{##1}%
6248   \glslabeltok{##2}%
6249   \glsshorttok{##3}%
6250   \glslongtok{##4}%
6251   \newacronymhook
6252   \DefaultNewAcronymDef
6253 }%
6254 \renewcommand*\acrpluralsuffix{\glspluralsuffix}%
6255 }

```

**\acrfootnote** Used by the footnote acronym styles.

```

6256 \newcommand*\acrfootnote}[3]{\acrlinkfootnote{#1}{#2}{#3}}

```

**\acrlinkfootnote**

```

6257 \newcommand*\acrlinkfootnote}[3]{%
6258   \footnote{\glslink{#1}{#2}{#3}}%
6259 }

```

**\acrnoflinkfootnote**

```

6260 \newcommand*\acrnoflinkfootnote}[3]{%
6261   \footnote{#3}%
6262 }

```

AcronymDisplayStyle Sets the acronym display style for given glossary for the description and footnote combination.

```

6263 \newcommand*{\SetDescriptionFootnoteAcronymDisplayStyle}[1]{%
6264   \def\glsentryfmt[#1]{%

6265     \ifdefempty\glscustomtext
6266     {%
6267       \ifglssused{\glslabel}%
6268       {%
6269         \acronymfont{\glsentryfmt}%
6270       }%
6271     {%
6272       \firstacronymfont{\glsentryfmt}%
6273       \ifglsshassymbol{\glslabel}%
6274       {%
6275         \expandafter\protect\expandafter\acrfootnote\expandafter
6276         {\@gls@link@opts}{\@gls@link@label}%
6277       {%
6278         \glsifplural
6279         {\glsentrysymbolplural{\glslabel}}%
6280         {\glsentrysymbol{\glslabel}}%
6281       }%
6282     }%
6283   }%
6284 }%
6285 {\glscustomtext\glsinsert}%
6286 }%
6287 }

```

otnoteNewAcronymDef

```

6288 \newcommand*{\DescriptionFootnoteNewAcronymDef}{%
6289   \edef\@do@newglossaryentry{%
6290     \noexpand\newglossaryentry{\the\glslabeltok}%
6291     {%
6292       type=\acronymtype,%
6293       name={\noexpand\acronymfont{\the\glsshorttok}},%
6294       sort={\the\glsshorttok},%
6295       first={\the\glsshorttok},%
6296       firstplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6297       text={\the\glsshorttok},%
6298       plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6299       short={\the\glsshorttok},%
6300       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6301       long={\the\glslongtok},%
6302       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6303       symbol={\the\glslongtok},%
6304       symbolplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6305       \the\glskeylisttok
6306     }%

```

```

6307 }%
6308 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6309 \let\@org@gls@assign@plural\gls@assign@plural
6310 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6311 \def\gls@assign@firstpl##1##2{%
6312   \@@gls@expand@field{##1}{firstpl}{##2}%
6313 }%
6314 \def\gls@assign@plural##1##2{%
6315   \@@gls@expand@field{##1}{plural}{##2}%
6316 }%
6317 \def\gls@assign@symbolplural##1##2{%
6318   \@@gls@expand@field{##1}{symbolplural}{##2}%
6319 }%
6320 \do@newglossaryentry
6321 \let\gls@assign@plural\@org@gls@assign@plural
6322 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6323 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6324 }

```

**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.

```

6325 \newcommand*{\SetDescriptionFootnoteAcronymStyle}{%
6326   \renewcommand{\newacronym}[4][[]]{%
6327     \ifx\@glsacronymlists\@empty
6328       \def\@glo@type{\acronymtype}%
6329       \setkeys{glossentry}{##1}%
6330       \DeclareAcronymList{\@glo@type}%
6331       \SetDescriptionFootnoteAcronymDisplayStyle{\@glo@type}%
6332     \fi
6333     \glskeylisttok{##1}%
6334     \glslabeltok{##2}%
6335     \glsshorttok{##3}%
6336     \glslongtok{##4}%
6337     \newacronymhook
6338     \DescriptionFootnoteNewAcronymDef
6339   }%

```

If footnote package option is specified, set the first use to append the long form (stored in symbol) as a footnote.

```

6340 \@for\@gls@type:=\@glsacronymlists\do{%
6341   \SetDescriptionFootnoteAcronymDisplayStyle{\@gls@type}%
6342 }%

```

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.

```

6343 \ifglsacrsmallcaps
6344   \renewcommand*{\acronymfont}[1]{\textsc{##1}}%

```

```

6345 \renewcommand*{\acrpluralsuffix}{%
6346 \glstextup{\glspluralsuffix}}%
6347 \else
6348 \ifglsacrsmaller
6349 \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%
6350 \fi
6351 \fi

```

Check for package option clash

```

6352 \ifglsacrdua
6353 \PackageError{glossaries}{Option clash: ‘footnote’ and ‘dua’
6354 can’t both be set}{}%
6355 \fi
6356 }%

```

**AcronymDisplayStyle** Sets the acronym display style for given glossary with description and dua combination.

```

6357 \newcommand*{\SetDescriptionDUAAcronymDisplayStyle}[1]{%
6358 \defglsentryfmt[#1]{\glsgenentryfmt}%
6359 }

```

**ionDUANewAcronymDef**

```

6360 \newcommand*{\DescriptionDUANewAcronymDef}{%
6361 \edef\@do@newglossaryentry{%
6362 \noexpand\newglossaryentry{\the\glslabeltok}%
6363 {%
6364 type=\acronymtype,%
6365 name={\the\glslongtok},%
6366 sort={\the\glslongtok},%
6367 text={\the\glslongtok},%
6368 first={\the\glslongtok},%
6369 plural={\noexpand\expandonce\noexpand\@glo@longpl},%
6370 firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6371 short={\the\glsshorttok},%
6372 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6373 long={\the\glslongtok},%
6374 longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6375 symbol={\the\glsshorttok},%
6376 symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6377 \the\glskeylisttok
6378 }%
6379 }%
6380 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6381 \let\@org@gls@assign@plural\gls@assign@plural
6382 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6383 \def\gls@assign@firstpl##1##2{%
6384 \@gls@expand@field{##1}{firstpl}{##2}%
6385 }%
6386 \def\gls@assign@plural##1##2{%

```

```

6387 \@@gls@expand@field{##1}{plural}{##2}%
6388 }%
6389 \def\gls@assign@symbolplural##1##2{%
6390 \@@gls@expand@field{##1}{symbolplural}{##2}%
6391 }%
6392 \do@newglossaryentry
6393 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6394 \let\gls@assign@plural\@org@gls@assign@plural
6395 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6396 }

```

**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.

```

6397 \newcommand*\SetDescriptionDUAAcronymStyle{%
6398 \ifglsacrsmallcaps
6399 \PackageError{glossaries}{Option clash: 'smallcaps' and 'dua'
6400 can't both be set}{}%
6401 \else
6402 \ifglsacrsmaller
6403 \PackageError{glossaries}{Option clash: 'smaller' and 'dua'
6404 can't both be set}{}%
6405 \fi
6406 \fi
6407 \renewcommand{\newacronym}[4][[]]{%
6408 \ifx\@glsacronymlists\@empty
6409 \def\@glo@type{\acronymtype}%
6410 \setkeys{glossentry}{##1}%
6411 \DeclareAcronymList{\@glo@type}%
6412 \SetDescriptionDUAAcronymDisplayStyle{\@glo@type}%
6413 \fi
6414 \glskeylisttok{##1}%
6415 \glslabeltok{##2}%
6416 \glsshorttok{##3}%
6417 \glslongtok{##4}%
6418 \newacronymhook
6419 \DescriptionDUANewAcronymDef
6420 }%

```

Set display.

```

6421 \@for\@gls@type:=\@glsacronymlists\do{%
6422 \SetDescriptionDUAAcronymDisplayStyle{\@gls@type}%
6423 }%
6424 }%

```

**AcronymDisplayStyle** Sets the acronym display style for given glossary using the description setting (but not footnote or dua).

```

6425 \newcommand*\SetDescriptionAcronymDisplayStyle[1]{%
6426 \defglsentryfmt[#1]{%

```

```

6427 \ifdefempty\glscustomtext
6428 {%
6429 \ifglssused{\glslabel}%
6430 {%
    Move the inserted text outside of \acronymfont
6431 \let\gls@org@insert\glsinsert
6432 \let\glsinsert\@empty
6433 \acronymfont{\glsgenentryfmt}\gls@org@insert
6434 }%
6435 {%
6436 \glsgenentryfmt
6437 \ifglshassymbol{\glslabel}%
6438 {%
6439 \glsifplural
6440 {%
6441 \def\@glo@symbol{\glsentrysymbolsymbolplural{\glslabel}}%
6442 }%
6443 {%
6444 \def\@glo@symbol{\glsentrysymbolsymbol{\glslabel}}%
6445 }%
6446 \space(\protect\firstacronymfont
6447 {\glscapscase
6448 {\@glo@symbol}
6449 {\@glo@symbol}
6450 {\mfirstucMakeUppercase{\@glo@symbol}}})%
6451 }%
6452 }%
6453 }%
6454 }%
6455 {\glscustomtext\glsinsert}%
6456 }%
6457 }

```

ptionNewAcronymDef

```

6458 \newcommand*{\DescriptionNewAcronymDef}{%
6459 \edef\@do@newglossaryentry{%
6460 \noexpand\newglossaryentry{\the\glslabeltok}%
6461 {%
6462 type=\acronymtype,%
6463 name={\noexpand
6464 \acrnameformat{\the\glsshorttok}{\the\glslongtok}},%
6465 sort={\the\glsshorttok},%
6466 first={\the\glslongtok},%
6467 firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6468 text={\the\glsshorttok},%
6469 plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6470 short={\the\glsshorttok},%
6471 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6472 long={\the\glslongtok},%

```



```

6473     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6474     symbol={\noexpand\@glo@text},%
6475     symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6476     \the\glskeylisttok}%
6477 }%
6478 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6479 \let\@org@gls@assign@plural\gls@assign@plural
6480 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6481 \def\gls@assign@firstpl##1##2{%
6482   \@@gls@expand@field{##1}{firstpl}{##2}%
6483 }%
6484 \def\gls@assign@plural##1##2{%
6485   \@@gls@expand@field{##1}{plural}{##2}%
6486 }%
6487 \def\gls@assign@symbolplural##1##2{%
6488   \@@gls@expand@field{##1}{symbolplural}{##2}%
6489 }%
6490 \do@newglossaryentry
6491 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6492 \let\gls@assign@plural\@org@gls@assign@plural
6493 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6494 }

```

**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.

```

6495 \newcommand*{\SetDescriptionAcronymStyle}{%
6496   \renewcommand{\newacronym}[4][\]{%
6497     \ifx\@glsacronymlists\@empty
6498       \def\@glo@type{\acronymtype}%
6499       \setkeys{glossentry}{##1}%
6500       \DeclareAcronymList{\@glo@type}%
6501       \SetDescriptionAcronymDisplayStyle{\@glo@type}%
6502     \fi
6503     \glskeylisttok{##1}%
6504     \glslabeltok{##2}%
6505     \glsshorttok{##3}%
6506     \glslongtok{##4}%
6507     \newacronymhook
6508     \DescriptionNewAcronymDef
6509   }%

```

Set display.

```

6510   \@for\@gls@type:=\@glsacronymlists\do{%
6511     \SetDescriptionAcronymDisplayStyle{\@gls@type}%
6512   }%

```

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.

```

6513 \ifglsmallcaps
6514 \renewcommand{\acronymfont}[1]{\textsc{##1}}
6515 \renewcommand*{\acrpluralsuffix}{%
6516 \glstextup{\glspuralsuffix}}%
6517 \else
6518 \ifglsmaller
6519 \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%
6520 \fi
6521 \fi
6522 }%

```

**AcronymDisplayStyle** Sets the acronym display style for given glossary with footnote setting (but not description or dua).

```

6523 \newcommand*{\SetFootnoteAcronymDisplayStyle}[1]{%
6524 \defglentryfmt[#1]{%

```

```

6525 \ifdefempty\glscustomtext
6526 {%

```

Move the inserted text outside of \acronymfont

```

6527 \let\gls@org@insert\glsinsert
6528 \let\glsinsert\@empty
6529 \ifglused{\glslabel}%
6530 {%
6531 \acronymfont{\glsgenentryfmt}\gls@org@insert
6532 }%
6533 {%
6534 \firstacronymfont{\glsgenentryfmt}\gls@org@insert
6535 \ifglshaslong{\glslabel}%
6536 {%
6537 \expandafter\protect\expandafter\acrfootnote\expandafter
6538 {\@gls@link@opts}{\@gls@link@label}%
6539 {%
6540 \glsifplural
6541 {\glstrylongpl{\glslabel}}%
6542 {\glstrylong{\glslabel}}%
6543 }%
6544 }%
6545 }%
6546 }%
6547 }%
6548 {\glscustomtext\glsinsert}%
6549 }%
6550 }

```

**FootnoteNewAcronymDef**

```

6551 \newcommand*{\FootnoteNewAcronymDef}{%

```

```

6552 \edef\@do@newglossaryentry{%
6553   \noexpand\newglossaryentry{\the\glslabeltok}%
6554   {%
6555     type=\acronymtype,%
6556     name={\noexpand\acronymfont{\the\glsshorttok}},%
6557     sort={\the\glsshorttok},%
6558     text={\the\glsshorttok},%
6559     plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6560     first={\the\glsshorttok},%
6561     firstplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6562     short={\the\glsshorttok},%
6563     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6564     long={\the\glslongtok},%
6565     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6566     description={\the\glslongtok},%
6567     descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6568     \the\glskeylisttok
6569   }%
6570 }%
6571 \let\@org@gls@assign@plural\gls@assign@plural
6572 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6573 \let\@org@gls@assign@descplural\gls@assign@descplural
6574 \def\gls@assign@firstpl##1##2{%
6575   \@@gls@expand@field{##1}{firstpl}{##2}%
6576 }%
6577 \def\gls@assign@plural##1##2{%
6578   \@@gls@expand@field{##1}{plural}{##2}%
6579 }%
6580 \def\gls@assign@descplural##1##2{%
6581   \@@gls@expand@field{##1}{descplural}{##2}%
6582 }%
6583 \@do@newglossaryentry
6584 \let\gls@assign@plural\@org@gls@assign@plural
6585 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6586 \let\gls@assign@descplural\@org@gls@assign@descplural
6587 }

```

**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.

```

6588 \newcommand*\SetFootnoteAcronymStyle{%
6589   \renewcommand{\newacronym}[4][]{%
6590     \ifx\@glsacronymlists\empty
6591       \def\@glo@type{\acronymtype}%
6592       \setkeys{glossentry}{##1}%
6593       \DeclareAcronymList{\@glo@type}%
6594       \SetFootnoteAcronymDisplayStyle{\@glo@type}%
6595     \fi
6596     \glskeylisttok{##1}%

```

```

6597 \glslabeltok{##2}%
6598 \glsshorttok{##3}%
6599 \gslongtok{##4}%
6600 \newacronymhook
6601 \FootnoteNewAcronymDef
6602 }%

```

#### Set display

```

6603 \@for\@gls@type:=\@glsacronymlists\do{%
6604 \SetFootnoteAcronymDisplayStyle{\@gls@type}%
6605 }%

```

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.

```

6606 \ifglsacrsmallcaps
6607 \renewcommand*{\acronymfont}[1]{\textsc{##1}}%
6608 \renewcommand*{\acrpluralsuffix}{%
6609 \glstextup{\glspluralsuffix}}%
6610 \else
6611 \ifglsacrsmaller
6612 \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%
6613 \fi
6614 \fi

```

#### Check for option clash

```

6615 \ifglsacrdua
6616 \PackageError{glossaries}{Option clash: ‘footnote’ and ‘dua’
6617 can’t both be set}}{}%
6618 \fi
6619 }%

```

`\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.

```

6620 \DeclareRobustCommand*{\glsdoparenifnotempty}[2]{%
6621 \protected@edef\gls@tmp{#1}%
6622 \ifdefempty\gls@tmp
6623 {}%
6624 {%
6625 \ifx\gls@tmp\@gls@default@value
6626 \else
6627 \space (#2{#1})%
6628 \fi
6629 }%
6630 }

```

`AcronymDisplayStyle` Sets the acronym display style for given glossary where neither footnote nor description is required, but smallcaps or smaller specified.

```

6631 \newcommand*{\SetSmallAcronymDisplayStyle}[1]{%

```

```

6632 \def\glsentryfmt[#1]{%
6633     \ifdefempty\glscustomtext
6634     {%
        Move the inserted text outside of \acronymfont
6635         \let\gls@org@insert\glsinsert
6636         \let\glsinsert\@empty
6637         \ifglused{\glslabel}%
6638         {%
6639             \acronymfont{\glsentryfmt}\gls@org@insert
6640         }%
6641         {%
6642             \glsentryfmt
6643             \ifglshassymbol{\glslabel}%
6644             {%
6645                 \glsifplural
6646                 {%
6647                     \def\@glo@symbol{\glsentrysymbolplural{\glslabel}}%
6648                 }%
6649                 {%
6650                     \def\@glo@symbol{\glsentrysymbol{\glslabel}}%
6651                 }%
6652                 \space
6653                 (\glscapscase
6654                 {\firstacronymfont{\@glo@symbol}}%
6655                 {\firstacronymfont{\@glo@symbol}}%
6656                 {\firstacronymfont{\mfirstucMakeUppercase{\@glo@symbol}}})%
6657             }%
6658             {}%
6659         }%
6660     }%
6661     {\glscustomtext\glsinsert}%
6662 }%
6663 }

```

\SmallNewAcronymDef

```

6664 \newcommand*{\SmallNewAcronymDef}{%
6665     \edef\@do@newglossaryentry{%
6666         \noexpand\newglossaryentry{\the\glslabeltok}%
6667         {%
6668             type=\acronymtype,%
6669             name={\noexpand\acronymfont{\the\glsshorttok}},%
6670             sort={\the\glsshorttok},%
6671             text={\the\glsshorttok},%
        }
        Default to the short plural.
6672         plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6673         first={\the\glslongtok},%
    }

```

Default to the long plural.

```

6674     firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6675     short={\the\glsshorttok},%
6676     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6677     long={\the\glslongtok},%
6678     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6679     description={\noexpand\@glo@first},%
6680     descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6681     symbol={\the\glsshorttok},%

```

Default to the short plural.

```

6682     symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6683     \the\glskeylisttok
6684 }%
6685 }%
6686 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6687 \let\@org@gls@assign@plural\gls@assign@plural
6688 \let\@org@gls@assign@descplural\gls@assign@descplural
6689 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6690 \def\gls@assign@firstpl##1##2{%
6691   \@@gls@expand@field{##1}{firstpl}{##2}%
6692 }%
6693 \def\gls@assign@plural##1##2{%
6694   \@@gls@expand@field{##1}{plural}{##2}%
6695 }%
6696 \def\gls@assign@descplural##1##2{%
6697   \@@gls@expand@field{##1}{descplural}{##2}%
6698 }%
6699 \def\gls@assign@symbolplural##1##2{%
6700   \@@gls@expand@field{##1}{symbolplural}{##2}%
6701 }%
6702 \do@newglossaryentry
6703 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6704 \let\gls@assign@plural\@org@gls@assign@plural
6705 \let\gls@assign@descplural\@org@gls@assign@descplural
6706 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6707 }

```

`\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.

```

6708 \newcommand*\SetSmallAcronymStyle{%
6709   \renewcommand{\newacronym}[4][\]{%
6710     \ifx\@glsacronymlists\@empty
6711       \def\@glo@type{\acronymtype}%
6712       \setkeys{glossentry}{##1}%
6713       \DeclareAcronymList{\@glo@type}%
6714       \SetSmallAcronymDisplayStyle{\@glo@type}%
6715     \fi
6716     \glskeylisttok{##1}%

```

```

6717 \glslabeltok{##2}%
6718 \glsshorttok{##3}%
6719 \glslongtok{##4}%
6720 \newacronymhook
6721 \SmallNewAcronymDef
6722 }%

```

Change the display since first only contains long form.

```

6723 \@for\@gls@type:=\@glsacronymlists\do{%
6724 \SetSmallAcronymDisplayStyle{\@gls@type}%
6725 }%

```

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.

```

6726 \ifglsacrsmallcaps
6727 \renewcommand*{\acronymfont}[1]{\textsc{##1}}
6728 \renewcommand*{\acrpluralsuffix}{%
6729 \glstextup{\glspluralsuffix}}%
6730 \else
6731 \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}
6732 \fi

```

check for option clash

```

6733 \ifglsacrdua
6734 \ifglsacrsmallcaps
6735 \PackageError{glossaries}{Option clash: ‘smallcaps’ and ‘dua’
6736 can’t both be set}{}%
6737 \else
6738 \PackageError{glossaries}{Option clash: ‘smaller’ and ‘dua’
6739 can’t both be set}{}%
6740 \fi
6741 \fi
6742 }%

```

\SetDUADisplayStyle Sets the acronym display style for given glossary with dua setting.

```

6743 \newcommand*{\SetDUADisplayStyle}[1]{%
6744 \defglsentryfmt[#1]{\glsgenentryfmt}%
6745 }

```

\DUANewAcronymDef

```

6746 \newcommand*{\DUANewAcronymDef}{%
6747 \edef\@do@newglossaryentry{%
6748 \noexpand\newglossaryentry{\the\glslabeltok}%
6749 {%
6750 type=\acronymtype,%
6751 name={\the\glsshorttok},%
6752 text={\the\glslongtok},%
6753 first={\the\glslongtok},%
6754 plural={\noexpand\expandonce\noexpand\@glo@longpl},%

```

```

6755     firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6756     short={\the\glsshorttok},%
6757     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6758     long={\the\glslongtok},%
6759     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6760     description={\the\glslongtok},%
6761     descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6762     symbol={\the\glsshorttok},%
6763     symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6764     \the\glskeylisttok
6765 }%
6766 }%
6767 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6768 \let\@org@gls@assign@plural\gls@assign@plural
6769 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6770 \let\@org@gls@assign@descplural\gls@assign@descplural
6771 \def\gls@assign@firstpl##1##2{%
6772   \@@gls@expand@field{##1}{firstpl}{##2}%
6773 }%
6774 \def\gls@assign@plural##1##2{%
6775   \@@gls@expand@field{##1}{plural}{##2}%
6776 }%
6777 \def\gls@assign@symbolplural##1##2{%
6778   \@@gls@expand@field{##1}{symbolplural}{##2}%
6779 }%
6780 \def\gls@assign@descplural##1##2{%
6781   \@@gls@expand@field{##1}{descplural}{##2}%
6782 }%
6783 \do@newglossaryentry
6784 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6785 \let\gls@assign@plural\@org@gls@assign@plural
6786 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6787 \let\gls@assign@descplural\@org@gls@assign@descplural
6788 }

```

`\SetDUASStyle` Always expand acronyms.

```

6789 \newcommand*\SetDUASStyle{%
6790   \renewcommand{\newacronym}[4][[]]{%
6791     \ifx\@glsacronymlists\@empty
6792       \def\@glo@type{\acronymtype}%
6793       \setkeys{glossentry}{##1}%
6794       \DeclareAcronymList{\@glo@type}%
6795       \SetDUADisplayStyle{\@glo@type}%
6796     \fi
6797     \glskeylisttok{##1}%
6798     \glslabeltok{##2}%
6799     \glsshorttok{##3}%
6800     \glslongtok{##4}%
6801     \newacronymhook

```



```

6802 \DUANewAcronymDef
6803 }%
    Set the display
6804 \@for\@gls@type:=\@glsacronymlists\do{%
6805 \SetDUADisplayStyle{\@gls@type}%
6806 }%
6807 }

```

\SetAcronymStyle

```

6808 \newcommand*\SetAcronymStyle}{%
6809 \SetDefaultAcronymStyle
6810 \ifglsacrdescription
6811 \ifglsacrfootnote
6812 \SetDescriptionFootnoteAcronymStyle
6813 \else
6814 \ifglsacrdua
6815 \SetDescriptionDUAAcronymStyle
6816 \else
6817 \SetDescriptionAcronymStyle
6818 \fi
6819 \fi
6820 \else
6821 \ifglsacrfootnote
6822 \SetFootnoteAcronymStyle
6823 \else
6824 \ifthenelse{\boolean{glsacrsmalldcaps}\OR
6825 \boolean{glsacrsmaller}}{%
6826 }%
6827 \SetSmallAcronymStyle
6828 }%
6829 }%
6830 \ifglsacrdua
6831 \SetDUASyle
6832 \fi
6833 }%
6834 \fi
6835 \fi
6836 }

```

Set the acronym style according to the package options

```
6837 \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.

```

6838 \newcommand*{\SetCustomDisplayStyle}[1]{%
6839   \def\glsentryfmt[#1]{\glsentryfmt}%
6840 }

```

#### CustomAcronymFields

```

6841 \newcommand*{\CustomAcronymFields}{%
6842   name={\the\glsshorttok},%
6843   description={\the\glslongtok},%
6844   first={\noexpand\acrfullformat{\the\glslongtok}{\the\glsshorttok}},%
6845   firstplural={\noexpand\acrfullformat
6846     {\noexpand\glsentrylongpl{\the\glslabeltok}}%
6847     {\noexpand\glsentryshortpl{\the\glslabeltok}}},%
6848   text={\the\glsshorttok},%
6849   plural={\the\glsshorttok\noexpand\acrpluralsuffix}%
6850 }

```

#### CustomNewAcronymDef

```

6851 \newcommand*{\CustomNewAcronymDef}{%
6852   \protected@edef\@do@newglossaryentry{%
6853     \noexpand\newglossaryentry{\the\glslabeltok}%
6854     {%
6855       type=\acronymtype,%
6856       short={\the\glsshorttok},%
6857       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6858       long={\the\glslongtok},%
6859       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6860       user1={\the\glsshorttok},%
6861       user2={\the\glsshorttok\noexpand\acrpluralsuffix},%
6862       user3={\the\glslongtok},%
6863       user4={\the\glslongtok\noexpand\acrpluralsuffix},%
6864       \CustomAcronymFields,%
6865       \the\glskeylisttok
6866     }%
6867   }%
6868   \@do@newglossaryentry
6869 }

```

#### \SetCustomStyle

```

6870 \newcommand*{\SetCustomStyle}{%
6871   \renewcommand{\newacronym}[4][\]{%
6872     \ifx\@glsacronymlists\@empty
6873       \def\@glo@type{\acronymtype}%
6874       \setkeys{glossentry}{##1}%
6875       \DeclareAcronymList{\@glo@type}%
6876       \SetCustomDisplayStyle{\@glo@type}%
6877     \fi
6878     \glskeylisttok{##1}%
6879     \glslabeltok{##2}%

```

```

6880 \glsshorttok{##3}%
6881 \glslongtok{##4}%
6882 \newacronymhook
6883 \CustomNewAcronymDef
6884 }%

Set the display
6885 \@for\@gls@type:=\@glsacronymlists\do{%
6886 \SetCustomDisplayStyle{\@gls@type}%
6887 }%
6888 }

```

## 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.

```
6889 \RequirePackage{glossary-hypernav}
```

The styles that use list-like environments. These are not loaded if the `nolist` option is used:

```
6890 \@gls@loadlist
```

The styles that use the `longtable` environment. These are not loaded if the `no-long package` option is used.

```
6891 \@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.

```
6892 \@gls@loadsuper
```

The tree-like styles. These are not loaded if the `notree package` option is used.

```
6893 \@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.

```

6894 \ifx\@glossary@default@style\relax
6895 \else
6896 \setglossarystyle{\@glossary@default@style}
6897 \fi

```

## 1.19 Debugging Commands

`\showgloparent` `\showgloparent{<label>}`

```

6898 \newcommand*{\showgloparent}[1]{%
6899 \expandafter\show\csname glo@glsdetoklabel{#1}@parent\endcsname

```

6900 }

\showglolevel    \showglolevel{<label>}

```
6901 \newcommand*{\showglolevel}[1]{%
6902   \expandafter\show\csname glo@\glsdetoklabel{#1}@level\endcsname
6903 }
```

\showglotext    \showglotext{<label>}

```
6904 \newcommand*{\showglotext}[1]{%
6905   \expandafter\show\csname glo@\glsdetoklabel{#1}@text\endcsname
6906 }
```

\showgloplural    \showgloplural{<label>}

```
6907 \newcommand*{\showgloplural}[1]{%
6908   \expandafter\show\csname glo@\glsdetoklabel{#1}@plural\endcsname
6909 }
```

\showglofirst    \showglofirst{<label>}

```
6910 \newcommand*{\showglofirst}[1]{%
6911   \expandafter\show\csname glo@\glsdetoklabel{#1}@first\endcsname
6912 }
```

\showglofirstpl    \showglofirstpl{<label>}

```
6913 \newcommand*{\showglofirstpl}[1]{%
6914   \expandafter\show\csname glo@\glsdetoklabel{#1}@firstpl\endcsname
6915 }
```

\showglotype    \showglotype{<label>}

```
6916 \newcommand*{\showglotype}[1]{%
6917   \expandafter\show\csname glo@\glsdetoklabel{#1}@type\endcsname
6918 }
```

\showglocounter    \showglocounter{<label>}

```
6919 \newcommand*{\showglocounter}[1]{%
6920   \expandafter\show\csname glo@\glsdetoklabel{#1}@counter\endcsname
6921 }
```

\showglouser    \showglouser{<label>}

```
6922 \newcommand*{\showglouser}[1]{%
6923   \expandafter\show\csname glo@\glsdetoklabel{#1}@useri\endcsname
6924 }
```

\showglouserii    \showglouserii{<label>}

```
6925 \newcommand*{\showglouserii}[1]{%
6926   \expandafter\show\csname glo@\glsdetoklabel{#1}@userii\endcsname
6927 }
```

\showglouseriii    \showglouseriii{<label>}

```
6928 \newcommand*{\showglouseriii}[1]{%
6929   \expandafter\show\csname glo@\glsdetoklabel{#1}@useriii\endcsname
6930 }
```

\showglouseriv    \showglouseriv{<label>}

```
6931 \newcommand*{\showglouseriv}[1]{%
6932   \expandafter\show\csname glo@\glsdetoklabel{#1}@useriv\endcsname
6933 }
```

\showglouserv    \showglouserv{<label>}

```
6934 \newcommand*{\showglouserv}[1]{%
6935   \expandafter\show\csname glo@\glsdetoklabel{#1}@userv\endcsname
6936 }
```

\showglouservi    \showglouservi{<label>}

```
6937 \newcommand*{\showglouservi}[1]{%
6938   \expandafter\show\csname glo@\glsdetoklabel{#1}@uservi\endcsname
6939 }
```

\showgloname    \showgloname{<label>}

```
6940 \newcommand*{\showgloname}[1]{%
6941   \expandafter\show\csname glo@\glsdetoklabel{#1}@name\endcsname
6942 }
```

\showglodesc    \showglodesc{<label>}

```
6943 \newcommand*{\showglodesc}[1]{%
6944   \expandafter\show\csname glo@\glsdetoklabel{#1}@desc\endcsname
6945 }
```

\showglodescplural    \showglodescplural{<label>}

```
6946 \newcommand*{\showglodescplural}[1]{%
6947   \expandafter\show\csname glo@\glsdetoklabel{#1}@descplural\endcsname
6948 }
```

\showglosort    \showglosort{<label>}

```
6949 \newcommand*{\showglosort}[1]{%
6950   \expandafter\show\csname glo@\glsdetoklabel{#1}@sort\endcsname
6951 }
```

\showglosymbol    \showglosymbol{<label>}

```
6952 \newcommand*{\showglosymbol}[1]{%
6953   \expandafter\show\csname glo@\glsdetoklabel{#1}@symbol\endcsname
6954 }
```

`\showglosymbolplural`    `\showglosymbolplural{\langle label \rangle}`

```
6955 \newcommand*{\showglosymbolplural}[1]{%  
6956   \expandafter\show\csname glo@\glsdetoklabel{#1}@symbolplural\endcsname  
6957 }
```

`\showgloshort`    `\showgloshort{\langle label \rangle}`

```
6958 \newcommand*{\showgloshort}[1]{%  
6959   \expandafter\show\csname glo@\glsdetoklabel{#1}@short\endcsname  
6960 }
```

`\showglolong`    `\showglolong{\langle label \rangle}`

```
6961 \newcommand*{\showglolong}[1]{%  
6962   \expandafter\show\csname glo@\glsdetoklabel{#1}@long\endcsname  
6963 }
```

`\showgloindex`    `\showgloindex{\langle label \rangle}`

```
6964 \newcommand*{\showgloindex}[1]{%  
6965   \expandafter\show\csname glo@\glsdetoklabel{#1}@index\endcsname  
6966 }
```

`\showgloflag`    `\showgloflag{\langle label \rangle}`

```
6967 \newcommand*{\showgloflag}[1]{%  
6968   \expandafter\show\csname ifglo@\glsdetoklabel{#1}@flag\endcsname  
6969 }
```

`\showgloloclist`    `\showgloloclist{\langle label \rangle}`

```
6970 \newcommand*{\showgloloclist}[1]{%  
6971   \expandafter\show\csname glo@\glsdetoklabel{#1}@loclist\endcsname  
6972 }
```

`\showacronymlists` `\showacronymlists`

Show list of glossaries that have been flagged as a list of acronyms.

```
6973 \newcommand*{\showacronymlists}{%
6974   \show\@glsacronymlists
6975 }
```

`\showglossaries` `\showglossaries`

Show list of defined glossaries.

```
6976 \newcommand*{\showglossaries}{%
6977   \show\@glo@types
6978 }
```

`\showglossaryin` `\showglossaryin{<glossary-label>}`

Show the ‘in’ extension for the given glossary.

```
6979 \newcommand*{\showglossaryin}[1]{%
6980   \expandafter\show\csname @glo@type@#1@in\endcsname
6981 }
```

`\showglossaryout` `\showglossaryout{<glossary-label>}`

Show the ‘out’ extension for the given glossary.

```
6982 \newcommand*{\showglossaryout}[1]{%
6983   \expandafter\show\csname @glo@type@#1@out\endcsname
6984 }
```

`\showglossarytitle` `\showglossarytitle{<glossary-label>}`

Show the title for the given glossary.

```
6985 \newcommand*{\showglossarytitle}[1]{%
6986   \expandafter\show\csname @glo@type@#1@title\endcsname
6987 }
```

`\showglossarycounter` `\showglossarycounter{<glossary-label>}`

Show the counter for the given glossary.

```
6988 \newcommand*{\showglossarycounter}[1]{%
6989   \expandafter\show\csname @glo@type@#1@counter\endcsname
6990 }
```



```
\showglossaryentries \showglossaryentries{\glossary-label}
```

Show the list of entry labels for the given glossary.

```
6991 \newcommand*{\showglossaryentries}[1]{%
6992   \expandafter\show\csname glolist@#1\endcsname
6993 }
```

## 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.

```
6994 \csname ifglscpatible-2.07\endcsname
6995   \RequirePackage{glossaries-compatible-207}
6996 \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.

```
6997 \NeedsTeXFormat{LaTeX2e}
6998 \ProvidesPackage{glossaries-prefix}[2013/11/14 v4.0 (NLCT)]
```

Pass all options to glossaries:

```
6999 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{glossaries}}
```

Process options:

```
7000 \ProcessOptions
```

Load glossaries:

```
7001 \RequirePackage{glossaries}
```

Add the new keys:

```
7002 \define@key{glossentry}{prefixfirst}{\def\@glo@entryprefixfirst{#1}}%
7003 \define@key{glossentry}{prefixfirstplural}{\def\@glo@entryprefixfirstplural{#1}}%
7004 \define@key{glossentry}{prefix}{\def\@glo@entryprefix{#1}}%
7005 \define@key{glossentry}{prefixplural}{\def\@glo@entryprefixplural{#1}}%
```

Add them to \@gls@keymap:

```
7006 \appto\@gls@keymap{,%  
7007   {prefixfirst}{prefixfirst},%  
7008   {prefixfirstplural}{prefixfirstplural},%  
7009   {prefix}{prefix},%  
7010   {prefixplural}{prefixplural}}%  
7011 }
```

Set the default values:

```
7012 \appto\@newglossaryentryprehook{%  
7013   \def\@glo@entryprefix{}%  
7014   \def\@glo@entryprefixplural{}%  
7015   \let\@glo@entryprefixfirst\@gls@default@value  
7016   \let\@glo@entryprefixfirstplural\@gls@default@value  
7017 }
```

Set the assignment code:

```
7018 \appto\@newglossaryentryposthook{%  
7019   \gls@assign@field{}\@glo@label{prefix}\@glo@entryprefix}%  
7020   \gls@assign@field{}\@glo@label{prefixplural}\@glo@entryprefixplural}%  
7021 }
```

If prefixfirst has not been supplied, make it the same as prefix.

```
7021   \expandafter\gls@assign@field\expandafter  
7022     {\csname glo@\@glo@label @prefix\endcsname}\@glo@label{prefixfirst}%  
7023     {\@glo@entryprefixfirst}}%
```

If prefixfirstplural has not been supplied, make it the same as prefixplural.

```
7024   \expandafter\gls@assign@field\expandafter  
7025     {\csname glo@\@glo@label @prefixplural\endcsname}\@glo@label}%  
7026     {prefixfirstplural}\@glo@entryprefixfirstplural}}%  
7027 }
```

Define commands to access these fields:

glsentryprefixfirst

```
7028 \newcommand*\glsentryprefixfirst[1]{\csuse{glo@#1@prefixfirst}}
```

glsentryprefixfirstplural

```
7029 \newcommand*\glsentryprefixfirstplural[1]{\csuse{glo@#1@prefixfirstplural}}
```

\glsentryprefix

```
7030 \newcommand*\glsentryprefix[1]{\csuse{glo@#1@prefix}}
```

\glsentryprefixplural

```
7031 \newcommand*\glsentryprefixplural[1]{\csuse{glo@#1@prefixplural}}
```

Now for the initial upper case variants:

Glsentryprefixfirst

```
7032 \newrobustcmd*\Glsentryprefixfirst[1]{%  
7033   \protected@edef\@glo@text{\csname glo@#1@prefixfirst\endcsname}%  
7034   \xmakefirstuc\@glo@text  
7035 }
```

ryprefixfirstplural

```
7036 \newrobustcmd*{\Glsentryprefixfirstplural}[1]{%
7037   \protected@edef\@glo@text{\csname glo@#1@prefixfirstplural\endcsname}%
7038   \xmakefirstuc\@glo@text
7039 }
```

\Glsentryprefix

```
7040 \newrobustcmd*{\Glsentryprefix}[1]{%
7041   \protected@edef\@glo@text{\csname glo@#1@prefix\endcsname}%
7042   \xmakefirstuc\@glo@text
7043 }
```

lsentryprefixplural

```
7044 \newrobustcmd*{\Glsentryprefixplural}[1]{%
7045   \protected@edef\@glo@text{\csname glo@#1@prefixplural\endcsname}%
7046   \xmakefirstuc\@glo@text
7047 }
```

Define commands to determine if the prefix keys have been set:

\ifglshasprefix

```
7048 \newcommand*{\ifglshasprefix}[3]{%
7049   \ifcempty{glo@#1@prefix}%
7050   {#3}%
7051   {#2}%
7052 }
```

ifglshasprefixplural

```
7053 \newcommand*{\ifglshasprefixplural}[3]{%
7054   \ifcempty{glo@#1@prefixplural}%
7055   {#3}%
7056   {#2}%
7057 }
```

ifglshasprefixfirst

```
7058 \newcommand*{\ifglshasprefixfirst}[3]{%
7059   \ifcempty{glo@#1@prefixfirst}%
7060   {#3}%
7061   {#2}%
7062 }
```

asprefixfirstplural

```
7063 \newcommand*{\ifglshasprefixfirstplural}[3]{%
7064   \ifcempty{glo@#1@prefixfirstplural}%
7065   {#3}%
7066   {#2}%
7067 }
```

Define commands that insert the prefix before commands like \gls:

```
\pgls
7068 \newrobustcmd{\pgls}{\@ifstar\@spgls\@pgls}
```

\@spgls Starred version.

```
7069 \newcommand*{\@spgls}[2][\@pgls@{hyper=false,#1}{#2}]
```

\@pgls Unstarred version.

```
7070 \newcommand*{\@pgls}[2][\%
7071 \new@ifnextchar[%
7072 {\@pgls@{#1}{#2}}%
7073 {\@pgls@{#1}{#2}[]}%
7074 }
```

\@pgls@ Read in the final optional argument:

```
7075 \def\@pgls@#1#2[#3]{%
7076 \glsdoifexists{#2}%
7077 {%
7078 \ifglsused{#2}%
7079 {%
7080 \glsentryprefix{#2}%
7081 }%
7082 {%
7083 \glsentryprefixfirst{#2}%
7084 }%
7085 \@gls@{#1}{#2}[#3]%
7086 }%
7087 }
```

Similarly for the plural version:

```
\pglsp1
7088 \newrobustcmd{\pglsp1}{\@ifstar\@spglsp1\@pglsp1}
```

\@spglsp1 Starred version.

```
7089 \newcommand*{\@spglsp1}[2][\@pglsp1@{hyper=false,#1}{#2}]
```

\@pglsp1 Unstarred version.

```
7090 \newcommand*{\@pglsp1}[2][\%
7091 \new@ifnextchar[%
7092 {\@pglsp1@{#1}{#2}}%
7093 {\@pglsp1@{#1}{#2}[]}%
7094 }
```

\@pglsp1@ Read in the final optional argument:

```
7095 \def\@pglsp1@#1#2[#3]{%
7096 \glsdoifexists{#2}%
7097 {%
7098 \ifglsused{#2}%
```

```

7099    {%
7100      \glstryentryprefixplural{#2}%
7101    }%
7102    {%
7103      \glstryentryprefixfirstplural{#2}%
7104    }%
7105    \@glsp1@{#1}{#2}[#3]%
7106  }%
7107 }

```

Now for the first letter upper case versions:

\Pgls

```
7108 \newrobustcmd{\Pgls}{\@ifstar\@sPgls\@Pgls}
```

\@sPgls Starred version.

```
7109 \newcommand*{\@sPgls}[2][\@Pgls@{hyper=false,#1}{#2}]
```

\@Pgls Unstarred version.

```

7110 \newcommand*{\@Pgls}[2][\%
7111   \new@ifnextchar[\%
7112   {\@Pgls@{#1}{#2}}\%
7113   {\@Pgls@{#1}{#2}}\%
7114 ]

```

\@Pgls@ Read in the final optional argument:

```

7115 \def\@Pgls@#1#2[#3]{%
7116   \glstryentryprefix{#2}%
7117   {%
7118     \ifglstryused{#2}%
7119     {%
7120       \ifglstryhasprefix{#2}%
7121       {%
7122         \glstryentryprefix{#2}%
7123         \@glstry@{#1}{#2}[#3]%
7124       }%
7125       {\@glstry@{#1}{#2}[#3]}%
7126     }%
7127     {%
7128       \ifglstryhasprefixfirst{#2}%
7129       {%
7130         \glstryentryprefixfirst{#2}%
7131         \@glstry@{#1}{#2}[#3]%
7132       }%
7133       {\@glstry@{#1}{#2}[#3]}%
7134     }%
7135   }%
7136 }

```

Similarly for the plural version:

```

\Plspl
7137 \newrobustcmd{\Plspl}{\@ifstar\@sPlspl\@Plspl}

\@sPlspl  Starred version.
7138 \newcommand*{\@sPlspl}[2][\@Plspl@{hyper=false,#1}{#2}]

\@Plspl  Unstarred version.
7139 \newcommand*{\@Plspl}[2][\%
7140   \new@ifnextchar[\%
7141   {\@Plspl@{#1}{#2}}\%
7142   {\@Plspl@{#1}{#2}}\%
7143 ]

\@Plspl@  Read in the final optional argument:
7144 \def\@Plspl@#1#2[#3]{\%
7145   \glsdoifexists{#2}\%
7146   {\%
7147     \ifglused{#2}\%
7148     {\%
7149       \ifglshasprefixplural{#2}\%
7150       {\%
7151         \Glsentryprefixplural{#2}\%
7152         \@Plspl@{#1}{#2}[#3]\%
7153       }\%
7154       {\@Plspl@{#1}{#2}[#3]}\%
7155     }\%
7156     {\%
7157       \ifglshasprefixfirstplural{#2}\%
7158       {\%
7159         \Glsentryprefixfirstplural{#2}\%
7160         \@Plspl@{#1}{#2}[#3]\%
7161       }\%
7162       {\@Plspl@{#1}{#2}[#3]}\%
7163     }\%
7164   }\%
7165 }

```

Finally the all upper case versions:

```

\PGLS
7166 \newrobustcmd{\PGLS}{\@ifstar\@sPGLS\@PGLS}

\@sPGLS  Starred version.
7167 \newcommand*{\@sPGLS}[2][\@PGLS@{hyper=false,#1}{#2}]

```

\@PGLS Unstarred version.

```
7168 \newcommand*{\@PGLS}[2][\%  
7169 \new@ifnextchar[\  
7170 {\@PGLS@{#1}{#2}}%  
7171 {\@PGLS@{#1}{#2}[]}%  
7172 }
```

\@PGLS@ Read in the final optional argument:

```
7173 \def\@PGLS@#1#2[#3]{%  
7174 \glsdoifexists{#2}%  
7175 {%  
7176 \ifglsused{#2}%  
7177 {%  
7178 \mfirstucMakeUppercase{\glsentryprefix{#2}}%  
7179 }%  
7180 {%  
7181 \mfirstucMakeUppercase{\glsentryprefixfirst{#2}}%  
7182 }%  
7183 \@GLS@{#1}{#2}[#3]%  
7184 }%  
7185 }
```

Plural version:

\PGLSp1

```
7186 \newrobustcmd{\PGLSp1}{\@ifstar\@sPGLSp1\@PGLSp1}
```

\@sPGLSp1 Starred version.

```
7187 \newcommand*{\@sPGLSp1}[2][\@PGLSp1@{hyper=false,#1}{#2}]
```

\@PGLSp1 Unstarred version.

```
7188 \newcommand*{\@PGLSp1}[2][\%  
7189 \new@ifnextchar[\  
7190 {\@PGLSp1@{#1}{#2}}%  
7191 {\@PGLSp1@{#1}{#2}[]}%  
7192 }
```

\@PGLSp1@ Read in the final optional argument:

```
7193 \def\@PGLSp1@#1#2[#3]{%  
7194 \glsdoifexists{#2}%  
7195 {%  
7196 \ifglsused{#2}%  
7197 {%  
7198 \mfirstucMakeUppercase{\glsentryprefixplural{#2}}%  
7199 }%  
7200 {%  
7201 \mfirstucMakeUppercase{\glsentryprefixfirstplural{#2}}%  
7202 }%  
7203 \@GLSp1@{#1}{#2}[#3]%
```

```
7204 }%
7205 }
```

### 3 Mfirstuc Documented Code

```
7206 \NeedsTeXFormat{LaTeX2e}
7207 \ProvidesPackage{mfirstuc}[2013/11/04 v1.08 (NLCT)]
```

Requires etoolbox:

```
7208 \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`.

```
7209 \newif\if@glscs
7210 \newtoks\@glsmfirst
7211 \newtoks\@glsmrest
7212 \newrobustcmd*{\makefirstuc}[1]{%
7213   \def\gls@argi{#1}%
7214   \ifx\gls@argi\@empty

   If the argument is empty, do nothing.

7215   \else

7216   \def\@gls@tmp{\ #1}%
7217   \@onelevel@sanitize\@gls@tmp
7218   \expandafter\@gls@checkcs\@gls@tmp\relax\relax
7219   \if@glscs
7220     \@gls@getbody #1{}\@nil
7221     \ifx\@gls@rest\@empty
7222       \glsmakefirstuc{#1}%
7223     \else
7224       \expandafter\@gls@split\@gls@rest\@nil
7225       \ifx\@gls@first\@empty
7226         \glsmakefirstuc{#1}%
7227       \else
7228         \expandafter\@glsmfirst\expandafter{\@gls@first}%
7229         \expandafter\@glsmrest\expandafter{\@gls@rest}%
7230         \edef\@gls@domfirstuc{\noexpand\@gls@body
7231           {\noexpand\glsmakefirstuc\the\@glsmfirst}%
7232           \the\@glsmrest}%
7233         \@gls@domfirstuc
7234       \fi
```



```

7235     \fi
7236     \else
7237         \glsmakefirstuc{#1}%
7238     \fi
7239 \fi
7240 }

```

Put first argument in \@gls@first and second argument in \@gls@rest:

```

7241 \def\@gls@split#1#2\@nil{%
7242     \def\@gls@first{#1}\def\@gls@rest{#2}%
7243 }

7244 \def\@gls@checkcs#1 #2#3\relax{%
7245     \def\@gls@argi{#1}\def\@gls@argii{#2}%
7246     \ifx\@gls@argi\@gls@argii
7247         \@glscstrue
7248     \else
7249         \@glscsfalse
7250     \fi
7251 }

```

\@gls@makefirstuc Make first thing upper case:

```

7252 \def\@gls@makefirstuc#1{\mfirstucMakeUppercase #1}

```

mfirstucMakeUppercase Allow user to replace \MakeUppercase with another case changing command.

```

7253 \newcommand*{\mfirstucMakeUppercase}{\MakeUppercase}

```

\glsmakefirstuc Provide a user command to make it easier to customise.

```

7254 \newcommand*{\glsmakefirstuc}[1]{\@gls@makefirstuc{#1}}

```

Get the first grouped argument and stores in \@gls@body.

```

7255 \def\@gls@getbody#1#{\def\@gls@body{#1}\@gls@gobbletonil}

```

Scoup up everything to \@nil and store in \@gls@rest:

```

7256 \def\@gls@gobbletonil#1\@nil{\def\@gls@rest{#1}}

```

\xmakefirstuc Expand argument once before applying \makefirstuc (added v1.01).

```

7257 \newcommand*{\xmakefirstuc}[1]{%
7258     \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).

```

7259 \newrobustcmd*{\capitalisewords}[1]{%
7260     \def\gls@add@space{}%
7261     \mfu@capitalisewords#1 \@nil\mfu@endcap
7262 }

```

```

7263 \def\mfu@capitalisewords#1 #2\mfu@endcap{%
7264   \def\mfu@cap@first{#1}%
7265   \def\mfu@cap@second{#2}%
7266   \gls@add@space
7267   \makefirstuc{#1}%
7268   \def\gls@add@space{ }%
7269   \ifx\mfu@cap@second\@nnil
7270     \let\next\mfu@cap\mfu@noop
7271   \else
7272     \let\next\mfu@cap\mfu@capitalisewords
7273   \fi
7274   \next\mfu@cap#2\mfu@endcap
7275 }
7276 \def\mfu@noop#1\mfu@endcap{}

```

`\xcapitalisewords` Short-cut command:

```

7277 \newcommand*\xcapitalisewords[1]{%
7278   \expandafter\capitalisewords\expandafter{#1}%
7279 }

```

## 4 Glossary Styles

### 4.1 Glossary hyper-navigation definitions (glossary-hypernav package)

Package Definition:

```

7280 \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`

```

7281 \newcommand*\glsnavhyperlink[3][\@glo@type]{%
7282   \edef\gls@grplabel{#2}\protected@edef\@gls@grptitle{#3}%
7283   \@glslink{glsn:#1@#2}{#3}}

```

`\glsnavhypertarget[⟨type⟩]{⟨label⟩}{⟨text⟩}`

This command makes `⟨text⟩` a hypertarget for the glossary group whose label is given by `⟨label⟩` in the glossary given by `⟨type⟩`. If `⟨type⟩` is omitted, `\@glo@type` is used which is set by `\printglossary` to the current glossary label.

```

\glsnavhypertarget
7284 \newcommand*{\glsnavhypertarget}[3][\@glo@type]{%
  Add this group to the aux file for re-run check.
7285   \protected@write\@auxout{}\string\@gls@hypergroup{#1}{#2}}%
  Add the target.
7286   \@glstarget{glsn:#1@#2}{#3}%
  Check list of know groups to determine if a re-run is required.
7287   \expandafter\let
7288     \expandafter\@gls@list\csname @gls@hypergroup@list@#1\endcsname
  Iterate through list and terminate loop if this group is found.
7289   \@for\@gls@elem:=\@gls@list\do{%
7290     \ifthenelse{\equal{\@gls@elem}{#2}}{\@endfortrue}{}}%
  Check if list terminated prematurely.
7291   \if@endfor
7292   \else
    This group was not included in the list, so issue a warning.
7293     \GlossariesWarningNoLine{Navigation panel
7294       for glossary type ‘#1’^^Jmissing group ‘#2’}%
7295     \gdef\gls@hypergroup@rerun{%
7296       \GlossariesWarningNoLine{Navigation panel
7297         has changed. Rerun LaTeX}}%
7298   \fi
7299 }

\gls@hypergroup@rerun Give a warning at the end if re-run required
7300 \let\gls@hypergroup@rerun\relax
7301 \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.
7302 \newcommand*{\@gls@hypergroup}[2]{%
7303 \ifundefined{\@gls@hypergroup@list@#1}{%
7304   \expandafter\xdef\csname @gls@hypergroup@list@#1\endcsname{#2}%
7305 }{%
7306   \expandafter\let\expandafter\@gls@tmp
7307     \csname @gls@hypergroup@list@#1\endcsname
7308   \expandafter\xdef\csname @gls@hypergroup@list@#1\endcsname{%
7309     \@gls@tmp,#2}%
7310 }%
7311 }

```

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`

```
7312 \newcommand*\glsnavigation{%
7313 \def\@gls@between{%
7314 \@ifundefined{\@gls@hypergroup@list@{\@glo@type}}{%
7315   \def\@gls@list{%
7316 }{%
7317   \expandafter\let\expandafter\@gls@list
7318     \csname \@gls@hypergroup@list@{\@glo@type}\endcsname
7319 }%
7320 \@for\@gls@tmp:=\@gls@list\do{%
7321   \@gls@between

7322   \@gls@getgrouptitle{\@gls@tmp}{\@gls@grptitle}%
7323   \glsnavhyperlink{\@gls@tmp}{\@gls@grptitle}%
7324   \let\@gls@between\glshypernavsep%
7325 }%
7326 }
```

`\glshypernavsep` Separator for the hyper navigation bar.

```
7327 \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`

```
7328 \newcommand*\glssymbolnav{%
7329 \glsnavhyperlink{glssymbols}{\glsgetgrouptitle{glssymbols}}%
7330 \glshypernavsep
7331 \glsnavhyperlink{glsnumbers}{\glsgetgrouptitle{glsnumbers}}%
7332 \glshypernavsep
7333 }
```

## 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.

```
7334 \ProvidesPackage{glossary-inline}[2013/11/14 v4.0 (NLCT)]
```

`inline` Define the inline style.

```
7335 \newglossarystyle{inline}{%
```

Start of glossary sets up first empty separator between entries. (This is then changed by `\glossentry`)

```
7336 \renewenvironment{theglossary}%
7337   {%
7338     \def\gls@inlinesep{}%
7339     \def\gls@inlinesubsep{}%
7340     \def\gls@inlinepostchild{}%
7341   }%
7342   {\glspostinline}%
```

No header:

```
7343 \renewcommand*{\glossaryheader}{}%
```

No group headings (if heading is required, add `\glsinlinedopostchild` to start definition in case heading follows a child entry):

```
7344 \renewcommand*{\glsgroupheading}[1]{}%
```

Just display separator followed by name and description:

```
7345 \renewcommand{\glossentry}[2] {%
7346   \glsinlinedopostchild
7347   \gls@inlinesep
7348   \glsentryitem{##1}%
7349   \glsinlinenameformat{##1}{%
7350     \glossentryname{##1}%
7351   }%
7352   \ifglshasdescsuppressed{##1}%
7353   {%
7354     \glsinlineemptydescformat
7355     {%
7356       \glossentrysymbol{##1}%
7357     }%
7358     {%
7359       ##2%
7360     }%
7361   }%
7362   {%
7363     \ifglshasdesc{##1}%
7364     {\glsinlinedescformat{\glossentrydesc{##1}}{\glossentrysymbol{##1}}{##2}}%
7365     {\glsinlineemptydescformat{\glossentrysymbol{##1}}{##2}}%
7366   }%
7367   \ifglshaschildren{##1}%
7368   {%
7369     \glsresetsubentrycounter
7370     \glsinlineparentchildseparator
7371     \def\gls@inlinesubsep{}%
7372     \def\gls@inlinepostchild{\glsinlinepostchild}%
7373   }%
7374   {}%
7375   \def\gls@inlinesep{\glsinlineseparator}%
7376 }%
```

Sub-entries display description:

```
7377 \renewcommand{\subglossentry}[3]{%
7378   \gls@inlinesubsep%
7379   \glsinlinesubnameformat{##2}{%
7380     \glossentryname{##2}}%
7381   \glssubentryitem{##2}%
7382   \glsinlinesubdescformat{\glossentrydesc{##2}}{\glossentrysymbol{##2}}{##3}%
7383   \def\gls@inlinesubsep{\glsinlinesubseparator}%
7384 }%
```

Nothing special between groups:

```
7385 \renewcommand*{\glsgroupskip}{}%
7386 }
```

`\glsinlinedopostchild`

```
7387 \newcommand*{\glsinlinedopostchild}{%
7388   \gls@inlinepostchild
7389   \def\gls@inlinepostchild{}%
7390 }
```

`\glsinlineseparator` Separator to use between entries.

```
7391 \newcommand*{\glsinlineseparator}{;\space}
```

`\glsinlinesubseparator` Separator to use between sub-entries.

```
7392 \newcommand*{\glsinlinesubseparator}{,\space}
```

`\glsinlinenparentchildseparator` Separator to use between parent and children.

```
7393 \newcommand*{\glsinlinenparentchildseparator}{:\space}
```

`\glsinlinepostchild` Hook to use between child and next entry

```
7394 \newcommand*{\glsinlinepostchild}{}
```

`\glsinlinepost` Terminator for inline glossary.

```
7395 \newcommand*{\glsinlinepost}{\glsinlinepostdescription\space}
```

`\glsinlinenameformat` Formats the name of the entry (first argument label, second argument name):

```
7396 \newcommand*{\glsinlinenameformat}[2]{\glstarget{#1}{#2}}
```

`\glsinlinedescformat` Formats the entry's description, symbol and location list:

```
7397 \newcommand*{\glsinlinedescformat}[3]{\space#1}
```

`\glsinlineemptydescformat` Formats the entry's symbol and location list when the description is empty:

```
7398 \newcommand*{\glsinlineemptydescformat}[2]{}
```

`\glsinlinesubnameformat` Formats the name of the subentry (first argument label, second argument name):

```
7399 \newcommand*{\glsinlinesubnameformat}[2]{\glstarget{#1}{}}
```

`\glsinlinesubdescformat` Formats the subentry's description, symbol and location list:

```
7400 \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.

```
7401 \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.

```
7402 \newglossarystyle{list}{%
```

Use description environment:

```
7403 \renewenvironment{theglossary}{%
```

```
7404 \begin{description}}{\end{description}}%
```

No header at the start of the environment:

```
7405 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7406 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries start a new item in the list:

```
7407 \renewcommand*{\glossentry}[2]{%
```

```
7408 \item[\glsentryitem{##1}]%
```

```
7409 \glstarget{##1}{\glossentryname{##1}}]
```

```
7410 \glossentrydesc{##1}\glspostdescription\space ##2}%
```

Sub-entries continue on the same line:

```
7411 \renewcommand*{\subglossentry}[3]{%
```

```
7412 \glssubentryitem{##2}]%
```

```
7413 \glstarget{##2}{\strut}]%
```

```
7414 \glossentrydesc{##2}\glspostdescription\space ##3.}%
```

```
7415% \end{macrocode}
```

```
7416% Add vertical space between groups:
```

```
7417%\changes{3.03}{2012/09/21}{added check for glsnogroupskip}
```

```
7418% \begin{macrocode}
```

```
7419 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
```

```
7420 }
```

**listgroup** The listgroup style is like the list style, but the glossary groups have headings.

```
7421 \newglossarystyle{listgroup}{%
```

Base it on the list style:

```
7422 \setglossarystyle{list}%
```

Each group has a heading:

```
7423 \renewcommand*{\glsgroupheading}[1]{\item[\glsgetgrouptitle{##1}]}
```

**listhypergroup** The listhypergroup style is like the listgroup style, but has a set of links to the groups at the start of the glossary.

```
7424 \newglossarystyle{listhypergroup}{%
```

Base it on the list style:

```
7425 \setglossarystyle{list}%
```

Add navigation links at the start of the environment:

```
7426 \renewcommand*{\glossaryheader}{%
```

```
7427 \item[\glsnavigation]}%
```

Each group has a heading with a hypertarget:

```
7428 \renewcommand*{\glsgroupheading}[1]{%
```

```
7429 \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.

```
7430 \newglossarystyle{altlist}{%
```

Base it on the list style:

```
7431 \setglossarystyle{list}%
```

Main (level 0) entries start a new item in the list with a line break after the entry name:

```
7432 \renewcommand*{\glossentry}[2]{%
```

```
7433 \item[\glsentryitem{##1}%
```

```
7434 \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.

```
7435 \mbox{}\par\nobreak\@afterheading
```

```
7436 \glossentrydesc{##1}\glspostdescription\space ##2}%
```

Sub-entries start a new paragraph:

```
7437 \renewcommand{\subglossentry}[3]{%
```

```
7438 \par
```

```
7439 \glssubentryitem{##2}%
```

```
7440 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space ##3}%
```

```
7441 }
```

**altlistgroup** The altlistgroup glossary style is like the altlist style, but the glossary groups have headings.

```
7442 \newglossarystyle{altlistgroup}{%
```

Base it on the altlist style:

```
7443 \setglossarystyle{altlist}%
```

Each group has a heading:

```
7444 \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.

```
7445 \newglossarystyle{altlisthypergroup}{%
```

Base it on the `altlist` style:

```
7446 \setglossarystyle{altlist}{%
```

Add navigation links at the start of the environment:

```
7447 \renewcommand*{\glossaryheader}{%
```

```
7448 \item[\glsnavigation]}%
```

Each group has a heading with a `hypertarget`:

```
7449 \renewcommand*{\glsgroupheading}[1]{%
```

```
7450 \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.

```
7451 \newglossarystyle{listdotted}{%
```

Base it on the `list` style:

```
7452 \setglossarystyle{list}{%
```

Each main (level 0) entry starts a new item:

```
7453 \renewcommand*{\glossentry}[2]{%
```

```
7454 \item[\makebox[\glslistdottedwidth][l]{%
```

```
7455 \glsentryitem{##1}%
```

```
7456 \glstarget{##1}{\glossentryname{##1}}%
```

```
7457 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}\glossentrydesc{##1}}%
```

Sub entries have the same format as main entries:

```
7458 \renewcommand*{\subglossentry}[3]{%
```

```
7459 \item[\makebox[\glslistdottedwidth][l]{%
```

```
7460 \glssubentryitem{##2}%
```

```
7461 \glstarget{##2}{\glossentryname{##2}}%
```

```
7462 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}\glossentrydesc{##2}}%
```

```
7463 }
```

`\glslistdottedwidth`

```
7464 \newlength\glslistdottedwidth
```

```
7465 \setlength{\glslistdottedwidth}{.5\hsize}
```

`sublistdotted` This style is similar to the `glostylelistdotted` style, except that the main entries just have the name displayed.

```
7466 \newglossarystyle{sublistdotted}{%
```

Base it on the `listdotted` style:

```
7467 \setglossarystyle{listdotted}{%
```

Main (level 0) entries just display the name:

```
7468 \renewcommand*{\glossentry}[2]{%
7469   \item[\glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}}}%
7470 }
```

## 4.4 Glossary Styles using longtable (the glossary-long package)

The glossary styles defined in the package used the longtable environment in the glossary.

```
7471 \ProvidesPackage{glossary-long}[2013/11/14 v4.0 (NLCT)]
```

Requires the package:

```
7472 \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`.)

```
7473 \@ifundefined{glsdescwidth}{%
7474   \newlength{glsdescwidth}
7475   \setlength{glsdescwidth}{0.6\hsize}
7476 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column.

```
7477 \@ifundefined{glspagelistwidth}{%
7478   \newlength{glspagelistwidth}
7479   \setlength{glspagelistwidth}{0.1\hsize}
7480 }{}
```

`long` The long glossary style command which uses the longtable environment:

```
7481 \newglossarystyle{long}{%
```

Use longtable with two columns:

```
7482   \renewenvironment{theglossary}%
7483     {\begin{longtable}\lp{glsdescwidth}}%
7484     {\end{longtable}}%
```

Do nothing at the start of the environment:

```
7485   \renewcommand*{\glossaryheader}{}%
```

No heading between groups:

```
7486   \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries displayed in a row:

```
7487   \renewcommand{\glossentry}[2]{%
7488     \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7489     \glossentrydesc{##1}\glspostdescription\space ##2\tabularnewline
7490   }%
```

Sub entries displayed on the following row without the name:

```
7491 \renewcommand{\subglossentry}[3]{%
7492     &
7493     \glssubentryitem{##2}%
7494     \glstarget{##2}{\strut}\glosentrydesc{##2}\glspostdescription\space
7495     ##3\tabularnewline
7496 }
```

Blank row between groups:

```
7497 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else &
7498 \tabularnewline\fi}%
7499 }
```

**longborder** The longborder style is like the above, but with horizontal and vertical lines:

```
7500 \newglossarystyle{longborder}{%
```

Base it on the glostylelong style:

```
7501 \setglossarystyle{long}%
```

Use longtable with two columns with vertical lines between each column:

```
7502 \renewenvironment{theglossary}{%
7503     \begin{longtable}{|l|p{\glsdescwidth}|}{\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
7504 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
7505 }
```

**longheader** The longheader style is like the long style but with a header:

```
7506 \newglossarystyle{longheader}{%
```

Base it on the glostylelong style:

```
7507 \setglossarystyle{long}%
```

Set the table's header:

```
7508 \renewcommand*{\glossaryheader}{%
7509     \bfseries \entryname & \bfseries \descriptionname\tabularnewline\endhead}%
7510 }
```

**longheaderborder** The longheaderborder style is like the long style but with a header and border:

```
7511 \newglossarystyle{longheaderborder}{%
```

Base it on the glostylelongborder style:

```
7512 \setglossarystyle{longborder}%
```

Set the table's header and add horizontal line to table's foot:

```
7513 \renewcommand*{\glossaryheader}{%
7514     \hline\bfseries \entryname & \bfseries
7515     \descriptionname\tabularnewline\hline
7516     \endhead
7517     \hline\endfoot}%
7518 }
```

**long3col** The long3col style is like long but with 3 columns

```
7519 \newglossarystyle{long3col}{%
```

Use a longtable with 3 columns:

```
7520 \renewenvironment{theglossary}{%
```

```
7521 {\begin{longtable}{lp{\glstdescwidth}p{\glspagelistwidth}}}%
```

```
7522 {\end{longtable}}}%
```

No table header:

```
7523 \renewcommand*{\glossaryheader}{}%
```

No headings between groups:

```
7524 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
7525 \renewcommand{\glossentry}[2]{%
```

```
7526 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
```

```
7527 \glossentrydesc{##1} & ##2\tabularnewline
```

```
7528 }%
```

Sub-entries on a separate row (no name, description in second column, page list in third column):

```
7529 \renewcommand{\subglossentry}[3]{%
```

```
7530 &
```

```
7531 \glssubentryitem{##2}%
```

```
7532 \glstarget{##2}{\strut}\glossentrydesc{##2} &
```

```
7533 ##3\tabularnewline
```

```
7534 }%
```

Blank row between groups:

```
7535 \renewcommand*{\glsgroupskip}{%
```

```
7536 \ifglsnogroupskip\else & &\tabularnewline\fi}%
```

```
7537 }
```

**long3colborder** The long3colborder style is like the long3col style but with a border:

```
7538 \newglossarystyle{long3colborder}{%
```

Base it on the glostylelong3col style:

```
7539 \setglossarystyle{long3col}%
```

Use a longtable with 3 columns with vertical lines around them:

```
7540 \renewenvironment{theglossary}{%
```

```
7541 {\begin{longtable}{|lp{\glstdescwidth}|p{\glspagelistwidth}|}}%
```

```
7542 {\end{longtable}}}%
```

Place horizontal lines at the head and foot of the table:

```
7543 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
```

```
7544 }
```

**long3colheader** The long3colheader style is like long3col but with a header row:

```
7545 \newglossarystyle{long3colheader}{%
```

Base it on the `glostylelong3col` style:

```
7546 \setglossarystyle{long3col}%
```

Set the table's header:

```
7547 \renewcommand*{\glossaryheader}{%
7548   \bfseries\entryname&\bfseries\descriptionname&
7549   \bfseries\pagelistname\tabularnewline\endhead}%
7550 }
```

`long3colheaderborder` The `long3colheaderborder` style is like the above but with a border

```
7551 \newglossarystyle{long3colheaderborder}{%
```

Base it on the `glostylelong3colborder` style:

```
7552 \setglossarystyle{long3colborder}%
```

Set the table's header and add horizontal line at table's foot:

```
7553 \renewcommand*{\glossaryheader}{%
7554   \hline
7555   \bfseries\entryname&\bfseries\descriptionname&
7556   \bfseries\pagelistname\tabularnewline\hline\endhead
7557   \hline\endfoot}%
7558 }
```

`long4col` The `long4col` style has four columns where the third column contains the value of the associated symbol key.

```
7559 \newglossarystyle{long4col}{%
```

Use a `longtable` with 4 columns:

```
7560 \renewenvironment{theglossary}%
7561   {\begin{longtable}{llll}}%
7562   {\end{longtable}}%
```

No table header:

```
7563 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7564 \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):

```
7565 \renewcommand{\glossentry}[2]{%
7566   \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7567   \glossentrydesc{##1} &
7568   \glossentrysymbol{##1} &
7569   ##2\tabularnewline
7570 }%
```

Sub entries on a single row with no name (description in second column, symbol in third column, page list in last column):

```
7571 \renewcommand{\subglossentry}[3]{%
7572   &
```

```

7573     \glssubentryitem{##2}%
7574     \glstarget{##2}{\strut}\glosseentrydesc{##2} &
7575     \glosseentrysymbol{##2} & ##3\tabularnewline
7576 }%

Blank row between groups:

7577 \renewcommand*{\glsgroupskip}{%
7578     \ifglsgnigroupskip\else & & \tabularnewline\fi}%
7579 }

```

**long4colheader** The long4colheader style is like long4col but with a header row.

```

7580 \newglossarystyle{long4colheader}{%

Base it on the glostylelong4col style:

7581 \setglossarystyle{long4col}%

Table has a header:

7582 \renewcommand*{\glossaryheader}{%
7583     \bfseries\entryname&\bfseries\descriptionname&
7584     \bfseries \symbolname&
7585     \bfseries\pagelistname\tabularnewline\endhead}%
7586 }

```

**long4colborder** The long4colborder style is like long4col but with a border.

```

7587 \newglossarystyle{long4colborder}{%

Base it on the glostylelong4col style:

7588 \setglossarystyle{long4col}%

Use a longtable with 4 columns surrounded by vertical lines:

7589 \renewenvironment{theglossary}%
7590     {\begin{longtable}{|l|l|l|l|}}%
7591     {\end{longtable}}%

Add horizontal lines to the head and foot of the table:

7592 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
7593 }

```

**long4colheaderborder** The long4colheaderborder style is like the above but with a border.

```

7594 \newglossarystyle{long4colheaderborder}{%

Base it on the glostylelong4col style:

7595 \setglossarystyle{long4col}%

Use a longtable with 4 columns surrounded by vertical lines:

7596 \renewenvironment{theglossary}%
7597     {\begin{longtable}{|l|l|l|l|}}%
7598     {\end{longtable}}%

Add table header and horizontal line at the table's foot:

7599 \renewcommand*{\glossaryheader}{%
7600     \hline\bfseries\entryname&\bfseries\descriptionname&

```

```

7601 \bfseries \symbolname&
7602 \bfseries\pagelistname\tabularnewline\hline\endhead
7603 \hline\endfoot}%
7604 }

```

**altlong4col** The altlong4col style is like the long4col style but can have multiline descriptions and page lists.

```

7605 \newglossarystyle{altlong4col}{%
    Base it on the glostylelong4col style:
7606 \setglossarystyle{long4col}%
    Use a longtable with 4 columns where the second and last columns may have
    multiple lines in each row:
7607 \renewenvironment{theglossary}%
7608 {\begin{longtable}{lp{\glsgdescwidth}lp{\glspagelistwidth}}}%
7609 {\end{longtable}}%
7610 }

```

**altlong4colheader** The altlong4colheader style is like altlong4col but with a header row.

```

7611 \newglossarystyle{altlong4colheader}{%
    Base it on the glostylelong4colheader style:
7612 \setglossarystyle{long4colheader}%
    Use a longtable with 4 columns where the second and last columns may have
    multiple lines in each row:
7613 \renewenvironment{theglossary}%
7614 {\begin{longtable}{lp{\glsgdescwidth}lp{\glspagelistwidth}}}%
7615 {\end{longtable}}%
7616 }

```

**altlong4colborder** The altlong4colborder style is like altlong4col but with a border.

```

7617 \newglossarystyle{altlong4colborder}{%
    Base it on the glostylelong4colborder style:
7618 \setglossarystyle{long4colborder}%
    Use a longtable with 4 columns where the second and last columns may have
    multiple lines in each row:
7619 \renewenvironment{theglossary}%
7620 {\begin{longtable}{|lp{\glsgdescwidth}|lp{\glspagelistwidth}|}%
7621 {\end{longtable}}%
7622 }

```

**altlong4colheaderborder** The altlong4colheaderborder style is like the above but with a header as well as a border.

```

7623 \newglossarystyle{altlong4colheaderborder}{%
    Base it on the glostylelong4colheaderborder style:
7624 \setglossarystyle{long4colheaderborder}%

```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
7625 \renewenvironment{theglossary}%
7626   {\begin{longtable}{|l|p{\glsdescwidth}|l|p{\glspagelistwidth}|}%
7627   {\end{longtable}}%
7628 }
```

#### 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.

```
7629 \ProvidesPackage{glossary-longragged}[2013/11/14 v4.0 (NLCT)]
```

Requires the package:

```
7630 \RequirePackage{array}
```

Requires the package:

```
7631 \RequirePackage{longtable}
```

`\glsdescwidth` This is a length that governs the width of the description column. This may have already been defined.

```
7632 \@ifundefined{glsdescwidth}{%
7633   \newlength\glsdescwidth
7634   \setlength{\glsdescwidth}{0.6\hsize}
7635 }
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

```
7636 \@ifundefined{glspagelistwidth}{%
7637   \newlength\glspagelistwidth
7638   \setlength{\glspagelistwidth}{0.1\hsize}
7639 }
```

`longragged` The longragged glossary style is like the long but uses ragged right formatting for the description column.

```
7640 \newglossarystyle{longragged}{%
```

Use longtable with two columns:

```
7641 \renewenvironment{theglossary}%
7642   {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}}}%
7643   {\end{longtable}}%
```

Do nothing at the start of the environment:

```
7644 \renewcommand*{\glossaryheader}{}%
```

No heading between groups:

```
7645 \renewcommand*{\glsgroupheading}[1]{}%
```



Main (level 0) entries displayed in a row:

```

7646 \renewcommand{\glossentry}[2]{%
7647   \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7648   \glossentrydesc{##1}\glspostdescription\space ##2%
7649   \tabularnewline
7650 }%

```

Sub entries displayed on the following row without the name:

```

7651 \renewcommand{\subglossentry}[3]{%
7652   &
7653   \glssubentryitem{##2}%
7654   \glstarget{##2}{\strut}\glossentrydesc{##2}%
7655   \glspostdescription\space ##3%
7656   \tabularnewline
7657 }%

```

Blank row between groups:

```

7658 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & \tabularnewline\fi}%
7659 }

```

**longraggedborder** The longraggedborder style is like the above, but with horizontal and vertical lines:

```

7660 \newglossarystyle{longraggedborder}{%

```

Base it on the glosstylelongragged style:

```

7661 \setglossarystyle{longragged}%

```

Use longtable with two columns with vertical lines between each column:

```

7662 \renewenvironment{theglossary}{%
7663   \begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|}%
7664   {\end{longtable}}%

```

Place horizontal lines at the head and foot of the table:

```

7665 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
7666 }

```

**longraggedheader** The longraggedheader style is like the longragged style but with a header:

```

7667 \newglossarystyle{longraggedheader}{%

```

Base it on the glosstylelongragged style:

```

7668 \setglossarystyle{longragged}%

```

Set the table's header:

```

7669 \renewcommand*{\glossaryheader}{%
7670   \bfseries \entryname & \bfseries \descriptionname
7671   \tabularnewline\endhead}%
7672 }

```

**raggedheaderborder** The longraggedheaderborder style is like the longragged style but with a header and border:

```

7673 \newglossarystyle{longraggedheaderborder}{%

```

Base it on the `glostylelongraggedborder` style:

```
7674 \setglossarystyle{longraggedborder}%
```

Set the table's header and add horizontal line to table's foot:

```
7675 \renewcommand*{\glossaryheader}{%  
7676 \hline\bfseries \entryname & \bfseries \descriptionname  
7677 \tabularnewline\hline  
7678 \endhead  
7679 \hline\endfoot}%  
7680 }
```

`longragged3col` The `longragged3col` style is like `longragged` but with 3 columns

```
7681 \newglossarystyle{longragged3col}{%
```

Use a longtable with 3 columns:

```
7682 \renewenvironment{theglossary}%  
7683 {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}%  
7684 >{\raggedright}p{\glspagelistwidth}}}%  
7685 {\end{longtable}}%
```

No table header:

```
7686 \renewcommand*{\glossaryheader}{}%
```

No headings between groups:

```
7687 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
7688 \renewcommand{\glossentry}[2]{%  
7689 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &  
7690 \glossentrydesc{##1} & ##2\tabularnewline  
7691 }%
```

Sub-entries on a separate row (no name, description in second column, page list in third column):

```
7692 \renewcommand{\subglossentry}[3]{%  
7693 &  
7694 \glssubentryitem{##2}%  
7695 \glstarget{##2}{\strut}\glossentrydesc{##2} &  
7696 ##3\tabularnewline  
7697 }%
```

Blank row between groups:

```
7698 \renewcommand*{\glsgroupskip}{%  
7699 \ifglsnogroupskip\else & &\tabularnewline\fi}%  
7700 }
```

`longragged3colborder` The `longragged3colborder` style is like the `longragged3col` style but with a border:

```
7701 \newglossarystyle{longragged3colborder}{%
```

Base it on the `glostylelongragged3col` style:

```
7702 \setglossarystyle{longragged3col}%
```

Use a `longtable` with 3 columns with vertical lines around them:

```
7703 \renewenvironment{theglossary}%  
7704   {\begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|%  
7705    >{\raggedright}p{\glspagelistwidth}|}%  
7706   {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
7707 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%  
7708 }
```

`ongragged3colheader` The `longragged3colheader` style is like `longragged3col` but with a header row:

```
7709 \newglossarystyle{longragged3colheader}{%
```

Base it on the `glostylelongragged3col` style:

```
7710 \setglossarystyle{longragged3col}%
```

Set the table's header:

```
7711 \renewcommand*{\glossaryheader}{%  
7712   \bfseries\entryname&\bfseries\descriptionname&  
7713   \bfseries\pagelistname\tabularnewline\endhead}%  
7714 }
```

`ged3colheaderborder` The `longragged3colheaderborder` style is like the above but with a border

```
7715 \newglossarystyle{longragged3colheaderborder}{%
```

Base it on the `glostylelongragged3colborder` style:

```
7716 \setglossarystyle{longragged3colborder}%
```

Set the table's header and add horizontal line at table's foot:

```
7717 \renewcommand*{\glossaryheader}{%  
7718   \hline  
7719   \bfseries\entryname&\bfseries\descriptionname&  
7720   \bfseries\pagelistname\tabularnewline\hline\endhead  
7721   \hline\endfoot}%  
7722 }
```

`altlongragged4col` The `altlongragged4col` style is like the `altlong4col` style defined in the package, except that ragged right formatting is used for the description and page list columns.

```
7723 \newglossarystyle{altlongragged4col}{%
```

Use a `longtable` with 4 columns where the second and last columns may have multiple lines in each row:

```
7724 \renewenvironment{theglossary}%  
7725   {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%  
7726    >{\raggedright}p{\glspagelistwidth}}}%  
7727   {\end{longtable}}%
```

No table header:

```
7728 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7729 \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):

```
7730 \renewcommand{\glossentry}[2]{%
7731   \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7732   \glossentrydesc{##1} & \glossentrydesc{##1} &
7733   ##2\tabularnewline
7734 }%
```

Sub entries on a single row with no name (description in second column, symbol in third column, page list in last column):

```
7735 \renewcommand{\subglossentry}[3]{%
7736   &
7737   \glssubentryitem{##2}%
7738   \glstarget{##2}{\strut}\glossentrydesc{##2} &
7739   \glossentrysymbol{##2} & ##3\tabularnewline
7740 }%
```

Blank row between groups:

```
7741 \renewcommand*{\glsgroupskip}{%
7742   \ifglisnogroupskip\else & & &\tabularnewline\fi}%
7743 }
```

`ongragged4colheader` The `altlongragged4colheader` style is like `altlongragged4col` but with a header row.

```
7744 \newglossarystyle{altlongragged4colheader}{%
```

Base it on the `glostylealtlongragged4col` style:

```
7745 \setglossarystyle{altlongragged4col}%
```

Use a `longtable` with 4 columns where the second and last columns may have multiple lines in each row:

```
7746 \renewenvironment{theglossary}%
7747   {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%
7748     >{\raggedright}p{\glspagelistwidth}}}%
7749   {\end{longtable}}%
```

Table has a header:

```
7750 \renewcommand*{\glossaryheader}{%
7751   \bfseries\entryname&\bfseries\descriptionname&
7752   \bfseries \symbolname&
7753   \bfseries\pagelistname\tabularnewline\endhead}%
7754 }
```

`ongragged4colborder` The `altlongragged4colborder` style is like `altlongragged4col` but with a border.

```
7755 \newglossarystyle{altlongragged4colborder}{%
```

Base it on the `glostylealtlongragged4col` style:

```
7756 \setglossarystyle{altlongragged4col}%
```

Use a `longtable` with 4 columns where the second and last columns may have multiple lines in each row:

```
7757 \renewenvironment{theglossary}%  
7758 {\begin{longtable}{|l|>{\raggedright}p{\glstdescwidth}|l|}%  
7759 >{\raggedright}p{\glspagelistwidth}|}%  
7760 {\end{longtable}}%
```

Add horizontal lines to the head and foot of the table:

```
7761 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%  
7762 }
```

`ged4colheaderborder` The `altlongragged4colheaderborder` style is like the above but with a header as well as a border.

```
7763 \newglossarystyle{altlongragged4colheaderborder}{%
```

Base it on the `glostylealtlongragged4col` style:

```
7764 \setglossarystyle{altlongragged4col}%
```

Use a `longtable` with 4 columns where the second and last columns may have multiple lines in each row:

```
7765 \renewenvironment{theglossary}%  
7766 {\begin{longtable}{|l|>{\raggedright}p{\glstdescwidth}|l|}%  
7767 >{\raggedright}p{\glspagelistwidth}|}%  
7768 {\end{longtable}}%
```

Add table header and horizontal line at the table's foot:

```
7769 \renewcommand*{\glossaryheader}{%  
7770 \hline\bfseries\entryname&\bfseries\descriptionname&  
7771 \bfseries \symbolname&  
7772 \bfseries\pagelistname\tabularnewline\hline\endhead  
7773 \hline\endfoot}%  
7774 }
```

## 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.

```
7775 \ProvidesPackage{glossary-mcols}[2013/11/14 v4.0 (NLCT)]
```

Required packages:

```
7776 \RequirePackage{multicol}  
7777 \RequirePackage{glossary-tree}
```

`\glsmcols` Define macro in which to store the number of columns. (Defaults to 2.)

```
7778 \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.)

```

7779 \newglossarystyle{mcolindex}{%
7780   \setglossarystyle{index}%
7781   \renewenvironment{theglossary}%
7782     {%

7783     \begin{multicols}{\glsmcols}
7784     \setlength{\parindent}{0pt}%
7785     \setlength{\parskip}{0pt plus 0.3pt}%
7786     \let\item\@idxitem}%
7787     {\end{multicols}}}%
7788 }
```

**mcolindexgroup** As mcolindex but has headings:

```

7789 \newglossarystyle{mcolindexgroup}{%
7790   \setglossarystyle{mcolindex}%
7791   \renewcommand*{\glsgroupeheading}[1]{%
7792     \item\textbf{\glsgrouptitle{##1}}\indexspace}%
7793 }
```

**mcolindexhypergroup** The mcolindexhypergroup style is like the mcolindexgroup style but has hyper navigation.

```

7794 \newglossarystyle{mcolindexhypergroup}{%

  Base it on the glostylemcolindex style:
7795   \setglossarystyle{mcolindex}%

  Put navigation links to the groups at the start of the glossary:
7796   \renewcommand*{\glossaryheader}{%
7797     \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.
7798   \renewcommand*{\glsgroupeheading}[1]{%
7799     \item\textbf{\glsnavhypertarget{##1}}{\glsgrouptitle{##1}}}%
7800     \indexspace}%
7801 }
```

**mcoltree** Multi-column index style. Same as the tree, but puts the glossary in multiple columns.

```

7802 \newglossarystyle{mcoltree}{%
7803   \setglossarystyle{tree}%
7804   \renewenvironment{theglossary}%
7805     {%

7806     \begin{multicols}{\glsmcols}
7807     \setlength{\parindent}{0pt}%
7808     \setlength{\parskip}{0pt plus 0.3pt}%

```

```

7809 }%
7810 {\end{multicols}}}%
7811 }

```

**mcoltreegroup** Like the mcoltree style but the glossary groups have headings.

```

7812 \newglossarystyle{mcoltreegroup}{%
    Base it on the glostylemcoltree style:
7813 \setglossarystyle{mcoltree}%
    Each group has a heading (in bold) followed by a vertical gap):
7814 \renewcommand{\glsgroupheading}[1]{\par
7815 \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
7816 }

```

**mcoltreehypergroup** The mcoltreehypergroup style is like the treegroup style, but has a set of links to the groups at the start of the glossary.

```

7817 \newglossarystyle{mcoltreehypergroup}{%
    Base it on the glostylemcoltree style:
7818 \setglossarystyle{mcoltree}%
    Put navigation links to the groups at the start of the theglossary environment:
7819 \renewcommand*{\glsgroupheading}{%
7820 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
    Each group has a heading (in bold with a target) followed by a vertical gap):
7821 \renewcommand*{\glsgroupheading}[1]{%
7822 \par\noindent
7823 \textbf{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}\par
7824 \indexspace}%
7825 }

```

**mcoltreenoname** Multi-column index style. Same as the treenoname, but puts the glossary in multiple columns.

```

7826 \newglossarystyle{mcoltreenoname}{%
7827 \setglossarystyle{treenoname}%
7828 \renewenvironment{theglossary}%
7829 {%
7830 \begin{multicols}{\glsmcols}
7831 \setlength{\parindent}{0pt}%
7832 \setlength{\parskip}{0pt plus 0.3pt}%
7833 }%
7834 {\end{multicols}}}%
7835 }

```

**mcoltreenonamegroup** Like the mcoltreenoname style but the glossary groups have headings.

```

7836 \newglossarystyle{mcoltreenonamegroup}{%
    Base it on the glostylemcoltreenoname style:
7837 \setglossarystyle{mcoltreenoname}%

```

Give each group a heading:

```
7838 \renewcommand{\glsgroupheading}[1]{\par
7839 \noindent\textbf{\glsgetgrouptitle{##1}}\par\indexspace}%
7840 }
```

`reenonamehypergroup` The `mcoltreenonamehypergroup` style is like the `mcoltreenonamegroup` style, but has a set of links to the groups at the start of the glossary.

```
7841 \newglossarystyle{mcoltreenonamehypergroup}{%
```

Base it on the `glostylemcoltreenoname` style:

```
7842 \setglossarystyle{mcoltreenoname}%
```

Put navigation links to the groups at the start of the `theglossary` environment:

```
7843 \renewcommand*{\glossaryheader}{%
7844 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
7845 \renewcommand*{\glsgroupheading}[1]{%
7846 \par\noindent
7847 \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
7848 \indexspace}%
7849 }
```

`mcolalttree` Multi-column index style. Same as the `alttree`, but puts the glossary in multiple columns.

```
7850 \newglossarystyle{mcolalttree}{%
7851 \setglossarystyle{alttree}%
7852 \renewenvironment{theglossary}%
7853 {%
7854 \begin{multicols}{\glsmcols}
7855 \def\@gls@prevlevel{-1}%
7856 \mbox{}\par
7857 }%
7858 {\par\end{multicols}}%
7859 }
```

`mcolalttreegroup` Like the `mcolalttree` style but the glossary groups have headings.

```
7860 \newglossarystyle{mcolalttreegroup}{%
```

Base it on the `glostylemcolalttree` style:

```
7861 \setglossarystyle{mcolalttree}%
```

Give each group a heading.

```
7862 \renewcommand{\glsgroupheading}[1]{\par
7863 \def\@gls@prevlevel{-1}%
7864 \hangindent0pt\relax
7865 \parindent0pt\relax
7866 \textbf{\glsgetgrouptitle{##1}}\par\indexspace}%
7867 }
```



`colalttreehypergroup` The `colalttreehypergroup` style is like the `colalttreegroup` style, but has a set of links to the groups at the start of the glossary.

```
7868 \newglossarystyle{colalttreehypergroup}{%
```

Base it on the `glostymcolalttree` style:

```
7869 \setglossarystyle{colalttree}{%
```

Put the navigation links in the header

```
7870 \renewcommand*{\glossaryheader}{%
```

```
7871 \par
```

```
7872 \def\@gls@prevlevel{-1}%
```

```
7873 \hangindent0pt\relax
```

```
7874 \parindent0pt\relax
```

```
7875 \textbf{\glsnavigation}\par\indexspace}%
```

Put a hypertarget at the start of each group

```
7876 \renewcommand*{\glsgroupheading}[1]{%
```

```
7877 \par
```

```
7878 \def\@gls@prevlevel{-1}%
```

```
7879 \hangindent0pt\relax
```

```
7880 \parindent0pt\relax
```

```
7881 \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
```

```
7882 \indexspace}}
```

## 4.7 Glossary Styles using supertabular environment (glossary-super package)

The glossary styles defined in the package use the supertabular environment.

```
7883 \ProvidesPackage{glossary-super}[2013/11/14 v4.0 (NLCT)]
```

Requires the package:

```
7884 \RequirePackage{supertabular}
```

`\glsdescwidth` This is a length that governs the width of the description column. This may already have been defined if has been loaded.

```
7885 \@ifundefined{glsdescwidth}{%
```

```
7886 \newlength\glsdescwidth
```

```
7887 \setlength{\glsdescwidth}{0.6\hsize}
```

```
7888 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined if has been loaded.

```
7889 \@ifundefined{glspagelistwidth}{%
```

```
7890 \newlength\glspagelistwidth
```

```
7891 \setlength{\glspagelistwidth}{0.1\hsize}
```

```
7892 }{}
```

`super` The `super` glossary style uses the supertabular environment (it uses lengths defined in the package.)

```
7893 \newglossarystyle{super}{%
```

Put the glossary in a supertabular environment with two columns and no head or tail:

```
7894 \renewenvironment{theglossary}%
7895   {\tablehead{}\tabletail{}}%
7896   \begin{supertabular}{lp{\glsdescwidth}}%
7897   {\end{supertabular}}%
```

Do nothing at the start of the table:

```
7898 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7899 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries put in a row (name in first column, description and page list in second column):

```
7900 \renewcommand{\glossentry}[2]{%
7901   \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7902   \glossentrydesc{##1}\glspostdescription\space ##2\tabularnewline
7903 }%
```

Sub entries put in a row (no name, description and page list in second column):

```
7904 \renewcommand{\subglossentry}[3]{%
7905   &
7906   \glssubentryitem{##2}%
7907   \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space
7908   ##3\tabularnewline
7909 }%
```

Blank row between groups:

```
7910 \renewcommand*{\glsgroupskip}{}%
7911   \ifglsnogroupskip\else & \tabularnewline\fi}%
7912 }
```

**superborder** The superborder style is like the above, but with horizontal and vertical lines:

```
7913 \newglossarystyle{superborder}{}%
```

Base it on the glostylesuper style:

```
7914 \setglossarystyle{super}{}%
```

Put the glossary in a supertabular environment with two columns and a horizontal line in the head and tail:

```
7915 \renewenvironment{theglossary}%
7916   {\tablehead{\hline}\tabletail{\hline}%
7917   \begin{supertabular}{|lp{\glsdescwidth}|}%
7918   {\end{supertabular}}%
7919 }
```

**superheader** The superheader style is like the super style, but with a header:

```
7920 \newglossarystyle{superheader}{}%
```

Base it on the glostylesuper style:

```
7921 \setglossarystyle{super}{}%
```

Put the glossary in a supertabular environment with two columns, a header and no tail:

```

7922 \renewenvironment{theglossary}%
7923   {\tablehead{\bfseries \entryname &
7924     \bfseries \descriptionname\tabularnewline}%
7925     \tabletail{}}%
7926   \begin{supertabular}{lp{\glstdescwidth}}%
7927   {\end{supertabular}}%
7928 }
```

**superheaderborder** The superheaderborder style is like the super style but with a header and border:

```

7929 \newglossarystyle{superheaderborder}{%
```

Base it on the glostylesuper style:

```

7930   \setglossarystyle{super}%
```

Put the glossary in a supertabular environment with two columns, a header and horizontal lines above and below the table:

```

7931   \renewenvironment{theglossary}%
7932     {\tablehead{\hline\bfseries \entryname &
7933       \bfseries \descriptionname\tabularnewline\hline}%
7934       \tabletail{\hline}
7935       \begin{supertabular}{|lp{\glstdescwidth}|}%
7936       {\end{supertabular}}%
7937 }
```

**super3col** The super3col style is like the super style, but with 3 columns:

```

7938 \newglossarystyle{super3col}{%
```

Put the glossary in a supertabular environment with three columns and no head or tail:

```

7939   \renewenvironment{theglossary}%
7940     {\tablehead{}\tabletail{}}%
7941     \begin{supertabular}{lp{\glstdescwidth}p{\glspagelistwidth}}%
7942     {\end{supertabular}}%
```

Do nothing at the start of the table:

```

7943   \renewcommand*{\glossaryheader}{}%
```

No group headings:

```

7944   \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```

7945   \renewcommand{\glossentry}[2]{%
7946     \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7947     \glossentrydesc{##1} & ##2\tabularnewline
7948   }%
```

Sub entries on a row (no name, description in second column, page list in last column):

```

7949 \renewcommand{\subglossentry}[3]{%
7950     &
7951     \glssubentryitem{##2}%
7952     \glstarget{##2}{\strut}\glossentrydesc{##2} &
7953     ##3\tabularnewline
7954 }%
```

Blank row between groups:

```

7955 \renewcommand*{\glsgroupskip}{%
7956     \ifglsgnোগroupskip\else & &\tabularnewline\fi}%
7957 }
```

**super3colborder** The super3colborder style is like the super3col style, but with a border:

```

7958 \newglossarystyle{super3colborder}{%
    Base it on the glostylesuper3col style:
7959     \setglossarystyle{super3col}%
    Put the glossary in a supertabular environment with three columns and a horizontal line in the head and tail:
7960     \renewenvironment{theglossary}%
7961         {\tablehead{\hline}\tabletail{\hline}%
7962         \begin{supertabular}{|l|p{\glsgdescwidth}|p{\glspagelistwidth}|}%
7963         {\end{supertabular}}%
7964 }
```

**super3colheader** The super3colheader style is like the super3col style but with a header row:

```

7965 \newglossarystyle{super3colheader}{%
    Base it on the glostylesuper3col style:
7966     \setglossarystyle{super3col}%
    Put the glossary in a supertabular environment with three columns, a header and no tail:
7967     \renewenvironment{theglossary}%
7968         {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
7969             \bfseries\pagelistname\tabularnewline}\tabletail{}}%
7970         \begin{supertabular}{lp{\glsgdescwidth}p{\glspagelistwidth}}%
7971         {\end{supertabular}}%
7972 }
```

**super3colheaderborder** The super3colheaderborder style is like the super3col style but with a header and border:

```

7973 \newglossarystyle{super3colheaderborder}{%
    Base it on the glostylesuper3colborder style:
7974     \setglossarystyle{super3colborder}%

```

Put the glossary in a supertabular environment with three columns, a header with horizontal lines and a horizontal line in the tail:

```

7975 \renewenvironment{theglossary}%
7976   {\tablehead{\hline
7977     \bfseries\entryname&\bfseries\descriptionname&
7978     \bfseries\pagelistname\tabularnewline\hline}%
7979   \tabletail{\hline}%
7980   \begin{supertabular}{|l|p{\glstdescwidth}|p{\glspagelistwidth}|}%
7981   {\end{supertabular}}%
7982 }
```

**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.

```

7983 \newglossarystyle{super4col}{%
```

Put the glossary in a supertabular environment with four columns and no head or tail:

```

7984 \renewenvironment{theglossary}%
7985   {\tablehead{}\tabletail{}}%
7986   \begin{supertabular}{|l|l|l|l|}%
7987   \end{supertabular}}%
```

Do nothing at the start of the table:

```

7988 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```

7989 \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:

```

7990 \renewcommand{\glossentry}[2]{%
7991   \glssentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7992   \glossentrydesc{##1} &
7993   \glossentrysymbol{##1} & ##3\tabularnewline
7994 }%
```

Sub entries on a row with no name, the description in the second column, symbol in third column and page list in last column:

```

7995 \renewcommand{\subglossentry}[3]{%
7996   &
7997   \glssubentryitem{##2}%
7998   \glstarget{##2}{\strut}\glossentrydesc{##2} &
7999   \glossentrysymbol{##2} & ##3\tabularnewline
8000 }%
```

Blank row between groups:

```

8001 \renewcommand*{\glsgroupskip}{}%
8002   \ifglsgroupskip\else & & \tabularnewline\fi}%
8003 }
```

**super4colheader** The super4colheader style is like the super4col but with a header row.

```
8004 \newglossarystyle{super4colheader}{%
    Base it on the glostylesuper4col style:
8005   \setglossarystyle{super4col}%
    Put the glossary in a supertabular environment with four columns, a header and
    no tail:
8006   \renewenvironment{theglossary}%
8007     {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
8008       \bfseries\symbolname &
8009       \bfseries\pagelistname\tabularnewline}%
8010     \tabletail{}}%
8011     \begin{supertabular}{|l|l|l|l|}%
8012     {\end{supertabular}}}%
8013 }
```

**super4colborder** The super4colborder style is like the super4col but with a border.

```
8014 \newglossarystyle{super4colborder}{%
    Base it on the glostylesuper4col style:
8015   \setglossarystyle{super4col}%
    Put the glossary in a supertabular environment with four columns and a hori-
    zontal line in the head and tail:
8016   \renewenvironment{theglossary}%
8017     {\tablehead{\hline}\tabletail{\hline}%
8018     \begin{supertabular}{|l|l|l|l|}%
8019     {\end{supertabular}}}%
8020 }
```

**super4colheaderborder** The super4colheaderborder style is like the super4col but with a header and border.

```
8021 \newglossarystyle{super4colheaderborder}{%
    Base it on the glostylesuper4col style:
8022   \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:
8023   \renewenvironment{theglossary}%
8024     {\tablehead{\hline\bfseries\entryname&\bfseries\descriptionname&
8025       \bfseries\symbolname &
8026       \bfseries\pagelistname\tabularnewline\hline}%
8027     \tabletail{\hline}%
8028     \begin{supertabular}{|l|l|l|l|}%
8029     {\end{supertabular}}}%
8030 }
```

**altsuper4col** The altsuper4col glossary style is like super4col but has provision for multiline descriptions.

```
8031 \newglossarystyle{altsuper4col}{%  
    Base it on the glostylesuper4col style:  
8032 \setglossarystyle{super4col}%  
    Put the glossary in a supertabular environment with four columns and no head  
    or tail:  
8033 \renewenvironment{theglossary}%  
8034     {\tablehead{}\tabletail{}}%  
8035     \begin{supertabular}{lp{\glsgdescwidth}lp{\glspagelistwidth}}}%  
8036     {\end{supertabular}}}%  
8037 }
```

**altsuper4colheader** The altsuper4colheader style is like the altsuper4col but with a header row.

```
8038 \newglossarystyle{altsuper4colheader}{%  
    Base it on the glostylesuper4colheader style:  
8039 \setglossarystyle{super4colheader}%  
    Put the glossary in a supertabular environment with four columns, a header and  
    no tail:  
8040 \renewenvironment{theglossary}%  
8041     {\tablehead{\bfseries\entryname&\bfseries\descriptionname&  
8042       \bfseries\symbolname &  
8043       \bfseries\pagelistname\tabularnewline}\tabletail{}}%  
8044     \begin{supertabular}{lp{\glsgdescwidth}lp{\glspagelistwidth}}}%  
8045     {\end{supertabular}}}%  
8046 }
```

**altsuper4colborder** The altsuper4colborder style is like the altsuper4col but with a border.

```
8047 \newglossarystyle{altsuper4colborder}{%  
    Base it on the glostylesuper4colborder style:  
8048 \setglossarystyle{super4colborder}%  
    Put the glossary in a supertabular environment with four columns and a hori-  
    zontal line in the head and tail:  
8049 \renewenvironment{theglossary}%  
8050     {\tablehead{\hline}\tabletail{\hline}%  
8051     \begin{supertabular}%  
8052       {\lllp{\glsgdescwidth}lllp{\glspagelistwidth}}}%  
8053     {\end{supertabular}}}%  
8054 }
```

**per4colheaderborder** The altsuper4colheaderborder style is like the altsuper4col but with a header and border.

```
8055 \newglossarystyle{altsuper4colheaderborder}{%
```

Base it on the `glostylessuper4colheaderborder` style:

```
8056 \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:

```
8057 \renewenvironment{theglossary}%  
8058   {\tablehead{\hline  
8059     \bfseries\entryname &  
8060     \bfseries\descriptionname &  
8061     \bfseries\symbolname &  
8062     \bfseries\pagelistname\tabularnewline\hline}%  
8063   \tabletail{\hline}%  
8064   \begin{supertabular}%  
8065     {\lllp{\glstdescwidth}\lllp{\glspagelistwidth}}}%  
8066   {\end{supertabular}}%  
8067 }
```

#### 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.

```
8068 \ProvidesPackage{glossary-superragged}[2013/11/14 v4.0 (NLCT)]
```

Requires the package:

```
8069 \RequirePackage{array}
```

Requires the package:

```
8070 \RequirePackage{supertabular}
```

`\glstdescwidth` This is a length that governs the width of the description column. This may already have been defined.

```
8071 \@ifundefined{glstdescwidth}{%  
8072   \newlength{glstdescwidth}  
8073   \setlength{glstdescwidth}{0.6\hsize}  
8074 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

```
8075 \@ifundefined{glspagelistwidth}{%  
8076   \newlength{glspagelistwidth}  
8077   \setlength{glspagelistwidth}{0.1\hsize}  
8078 }{}
```

`superragged` The `superragged` glossary style uses the `supertabular` environment.

```
8079 \newglossarystyle{superragged}{%
```



Put the glossary in a supertabular environment with two columns and no head or tail:

```
8080 \renewenvironment{theglossary}%
8081   {\tablehead{}\tabletail{}}%
8082   \begin{supertabular}{|l>{\raggedright}p{\glsgdescwidth}}}%
8083   {\end{supertabular}}%
```

Do nothing at the start of the table:

```
8084 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
8085 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries put in a row (name in first column, description and page list in second column):

```
8086 \renewcommand{\glossentry}[2]{%
8087   \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
8088   \glossentrydesc{##1}\glspostdescription\space ##2%
8089   \tabularnewline
8090 }%
```

Sub entries put in a row (no name, description and page list in second column):

```
8091 \renewcommand{\subglossentry}[3]{%
8092   &
8093   \glssubentryitem{##2}%
8094   \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space
8095   ##3%
8096   \tabularnewline
8097 }%
```

Blank row between groups:

```
8098 \renewcommand*{\glsgroupskip}{\ifglsgnigroupskip\else & \tabularnewline\fi}%
8099 }
```

**superraggedborder** The superraggedborder style is like the above, but with horizontal and vertical lines:

```
8100 \newglossarystyle{superraggedborder}{%
```

Base it on the glostylessuperragged style:

```
8101 \setglossarystyle{superragged}%
```

Put the glossary in a supertabular environment with two columns and a horizontal line in the head and tail:

```
8102 \renewenvironment{theglossary}%
8103   {\tablehead{\hline}\tabletail{\hline}}%
8104   \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}}|}%
8105   {\end{supertabular}}%
8106 }
```

**superraggedheader** The superraggedheader style is like the super style, but with a header:

```
8107 \newglossarystyle{superraggedheader}{%
```

Base it on the `glostylesuperragged` style:

```
8108 \setglossarystyle{superragged}%
```

Put the glossary in a `supertabular` environment with two columns, a header and no tail:

```
8109 \renewenvironment{theglossary}%  
8110 {\tablehead{\bfseries \entryname & \bfseries \descriptionname  
8111 \tabularnewline}%  
8112 \tabletail{}}%  
8113 \begin{supertabular}{1>\raggedright}p{\glsgdescwidth}}}%  
8114 {\end{supertabular}}%  
8115 }
```

`rraggedheaderborder` The `superraggedheaderborder` style is like the `superragged` style but with a header and border:

```
8116 \newglossarystyle{superraggedheaderborder}{%
```

Base it on the `glostylesuper` style:

```
8117 \setglossarystyle{superragged}%
```

Put the glossary in a `supertabular` environment with two columns, a header and horizontal lines above and below the table:

```
8118 \renewenvironment{theglossary}%  
8119 {\tablehead{\hline\bfseries \entryname &  
8120 \bfseries \descriptionname\tabularnewline\hline}%  
8121 \tabletail{\hline}  
8122 \begin{supertabular}{|1|>\raggedright}p{\glsgdescwidth}|}}}%  
8123 {\end{supertabular}}%  
8124 }
```

`superragged3col` The `superragged3col` style is like the `superragged` style, but with 3 columns:

```
8125 \newglossarystyle{superragged3col}{%
```

Put the glossary in a `supertabular` environment with three columns and no head or tail:

```
8126 \renewenvironment{theglossary}%  
8127 {\tablehead{} \tabletail{}}%  
8128 \begin{supertabular}{1>\raggedright}p{\glsgdescwidth}%  
8129 >\raggedright}p{\glspagelistwidth}}}%  
8130 {\end{supertabular}}%
```

Do nothing at the start of the table:

```
8131 \renewcommand*\glossaryheader{}%
```

No group headings:

```
8132 \renewcommand*\glsgroupheading[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
8133 \renewcommand{\glossentry}[2]{}%  
8134 \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
```

```

8135 \glossentrydesc{##1} &
8136 ##2\tabularnewline
8137 }%

```

Sub entries on a row (no name, description in second column, page list in last column):

```

8138 \renewcommand{\subglossentry}[3]{%
8139 &
8140 \glssubentryitem{##2}%
8141 \glstarget{##2}{\strut}\glossentrydesc{##2} &
8142 ##3\tabularnewline
8143 }%

```

Blank row between groups:

```

8144 \renewcommand*{\glsgroupskip}{\ifglsgnogroupskip\else & &\tabularnewline\fi}%
8145 }

```

**superragged3colborder** The `superragged3colborder` style is like the `superragged3col` style, but with a border:

```

8146 \newglossarystyle{superragged3colborder}{%

```

Base it on the `glostylesuperragged3col` style:

```

8147 \setglossarystyle{superragged3col}%

```

Put the glossary in a `supertabular` environment with three columns and a horizontal line in the head and tail:

```

8148 \renewenvironment{theglossary}%
8149 {\tablehead{\hline}\tabletail{\hline}%
8150 \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}|%
8151 >{\raggedright}p{\glspagelistwidth}|}%
8152 {\end{supertabular}}%
8153 }

```

**superragged3colheader** The `superragged3colheader` style is like the `superragged3col` style but with a header row:

```

8154 \newglossarystyle{superragged3colheader}{%

```

Base it on the `glostylesuperragged3col` style:

```

8155 \setglossarystyle{superragged3col}%

```

Put the glossary in a `supertabular` environment with three columns, a header and no tail:

```

8156 \renewenvironment{theglossary}%
8157 {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
8158 \bfseries\pagelistname\tabularnewline}\tabletail{}}%
8159 \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}%
8160 >{\raggedright}p{\glspagelistwidth}}}%
8161 {\end{supertabular}}%
8162 }

```

ght3colheaderborder The superragged3colheaderborder style is like the superragged3col style but with a header and border:

```
8163 \newglossarystyle{superragged3colheaderborder}{%
```

Base it on the glostylesuperragged3colborder style:

```
8164 \setglossarystyle{superragged3colborder}{%
```

Put the glossary in a supertabular environment with three columns, a header with horizontal lines and a horizontal line in the tail:

```
8165 \renewenvironment{theglossary}{%
8166   {\tablehead{\hline
8167     \bfseries\entryname&\bfseries\descriptionname&
8168     \bfseries\pagelistname\tabularnewline\hline}%
8169   \tabletail{\hline}%
8170   \begin{supertabular}{|l|>{\raggedright}p{\glsdescwidth}|%
8171     >{\raggedright}p{\glspagelistwidth}|}%
8172   {\end{supertabular}}}%
8173 }
```

altsuperragged4col The altsuperragged4col glossary style is like altsuper4col style in the package but uses ragged right formatting in the description and page list columns.

```
8174 \newglossarystyle{altsuperragged4col}{%
```

Put the glossary in a supertabular environment with four columns and no head or tail:

```
8175 \renewenvironment{theglossary}{%
8176   {\tablehead{}\tabletail}%
8177   \begin{supertabular}{|l>{\raggedright}p{\glsdescwidth}l%
8178     >{\raggedright}p{\glspagelistwidth}}}%
8179   {\end{supertabular}}%
```

Do nothing at the start of the table:

```
8180 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
8181 \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:

```
8182 \renewcommand{\glossentry}[2]{%
8183   \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
8184   \glossentrydesc{##1} &
8185   \glossentrysymbol{##1} & ##2\tabularnewline
8186 }%
```

Sub entries on a row with no name, the description in the second column, symbol in third column and page list in last column:

```
8187 \renewcommand{\subglossentry}[3]{%
8188   &
8189   \glssubentryitem{##2}%
8190   \glstarget{##2}{\strut}\glossentrydesc{##2} &
```

```

8191 \glossentrysymbol{##2} & ##3\tabularnewline
8192 }%

```

Blank row between groups:

```

8193 \renewcommand*{\glsnoglobskip}{\ifglsnogroupskip\else & & \tabularnewline\fi}%
8194 }

```

**altperragged4colheader** The `altperragged4colheader` style is like the `altperragged4col` style but with a header row.

```

8195 \newglossarystyle{altperragged4colheader}{%

```

Base it on the `glostylealtperragged4col` style:

```

8196 \setglossarystyle{altperragged4col}%

```

Put the glossary in a `supertabular` environment with four columns, a header and no tail:

```

8197 \renewenvironment{theglossary}%
8198 {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
8199 \bfseries\symbolname &
8200 \bfseries\pagelistname\tabularnewline}\tabletail{}}%
8201 \begin{supertabular}{1>{\raggedright}p{\glsdescwidth}1%
8202 >{\raggedright}p{\glspagelistwidth}}}%
8203 {\end{supertabular}}%
8204 }

```

**altperragged4colborder** The `altperragged4colborder` style is like the `altperragged4col` style but with a border.

```

8205 \newglossarystyle{altperragged4colborder}{%

```

Base it on the `glostylealtperragged4col` style:

```

8206 \setglossarystyle{altperragged4col}%

```

Put the glossary in a `supertabular` environment with four columns and a horizontal line in the head and tail:

```

8207 \renewenvironment{theglossary}%
8208 {\tablehead{\hline}\tabletail{\hline}%
8209 \begin{supertabular}%
8210 {\|1|>{\raggedright}p{\glsdescwidth}|1|%
8211 >{\raggedright}p{\glspagelistwidth}|}}%
8212 {\end{supertabular}}%
8213 }

```

**altperragged4colheaderborder** The `altperragged4colheaderborder` style is like the `altperragged4col` style but with a header and border.

```

8214 \newglossarystyle{altperragged4colheaderborder}{%

```

Base it on the `glostylealtperragged4col` style:

```

8215 \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:

```

8216 \renewenvironment{theglossary}%
8217   {\tablehead{\hline
8218     \bfseries\entryname &
8219     \bfseries\descriptionname &
8220     \bfseries\symbolname &
8221     \bfseries\pagelistname\tabularnewline\hline}%
8222   \tabletail{\hline}%
8223   \begin{supertabular}%
8224     {|||>{\raggedright}p{\glsdescwidth}|l|}%
8225     >{\raggedright}p{\glspagelistwidth}|}}%
8226   {\end{supertabular}}%
8227 }

```

## 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.

```

8228 \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.

```

8229 \newglossarystyle{index}{%

```

Set the paragraph indentation and skip and define `\item` to be the same as that used by the index:

```

8230 \renewenvironment{theglossary}%
8231   {\setlength{\parindent}{0pt}%
8232   \setlength{\parskip}{0pt plus 0.3pt}%
8233   \let\item\@idxitem}%
8234   {\par}%

```

Do nothing at the start of the environment:

```

8235 \renewcommand*{\glossaryheader}{}%

```

No group headers:

```

8236 \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.

```

8237 \renewcommand*{\glossentry}[2]{%
8238   \item\glstentryitem{##1}\textbf{\glstarget{##1}{\glossentryname{##1}}}%
8239   \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
8240   \space \glossentrydesc{##1}\glspostdescription\space ##2%
8241 }%

```

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.

```

8242 \renewcommand{\subglossentry}[3]{%
8243   \ifcase##1\relax
8244     % level 0
8245     \item
8246   \or
8247     % level 1
8248     \subitem
8249     \glssubentryitem{##2}%
8250   \else
8251     % all other levels
8252     \subsubitem
8253   \fi
8254   \textbf{\glstarget{##2}{\glossentryname{##2}}}%
8255   \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
8256   \space\glossentrydesc{##2}\glspostdescription\space ##3%
8257 }%
```

Vertical gap between groups is the same as that used by indices:

```

8258 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}}
```

**indexgroup** The `indexgroup` style is like the `index` style but has headings.

```

8259 \newglossarystyle{indexgroup}{%
```

Base it on the `glostyleindex` style:

```

8260 \setglossarystyle{index}%
```

Add a heading for each group. This puts the group's title in bold followed by a vertical gap.

```

8261 \renewcommand*{\glsgroupheading}[1]{%
8262   \item\textbf{\glsgrouptitle{##1}}\indexspace}%
8263 }
```

**indexhypergroup** The `indexhypergroup` style is like the `indexgroup` style but has hyper navigation.

```

8264 \newglossarystyle{indexhypergroup}{%
```

Base it on the `glostyleindex` style:

```

8265 \setglossarystyle{index}%
```

Put navigation links to the groups at the start of the glossary:

```

8266 \renewcommand*{\glossaryheader}{%
8267   \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.

```

8268 \renewcommand*{\glsgroupheading}[1]{%
8269   \item\textbf{\glsnavigationhypertarget{##1}{\glsgrouptitle{##1}}}%
```

```

8270 \indexspace}%
8271 }

```

tree The tree glossary style is similar in style to the index style, but can have arbitrary levels.

```

8272 \newglossarystyle{tree}{%

```

Set the paragraph indentation and skip:

```

8273 \renewenvironment{theglossary}%
8274 {\setlength{\parindent}{0pt}%
8275 \setlength{\parskip}{0pt plus 0.3pt}}%
8276 {}%

```

Do nothing at the start of the theglossary environment:

```

8277 \renewcommand*{\glossaryheader}{}%

```

No group headings:

```

8278 \renewcommand*{\glsgroupheading}[1]{}%

```

Main (level 0) entries: name in bold, followed by symbol in brackets (if it exists), the description and the page list:

```

8279 \renewcommand{\glossentry}[2]{%
8280 \hangindent0pt\relax
8281 \parindent0pt\relax
8282 \glstryitem{##1}\textbf{\glstarget{##1}{\glossentryname{##1}}}%
8283 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
8284 \space\glossentrydesc{##1}\glspostdescription\space##2\par
8285 }%

```

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.

```

8286 \renewcommand{\subglossentry}[3]{%
8287 \hangindent##1\glstreeindent\relax
8288 \parindent##1\glstreeindent\relax
8289 \ifnum##1=1\relax
8290 \glssubentryitem{##2}%
8291 \fi
8292 \textbf{\glstarget{##2}{\glossentryname{##2}}}%
8293 \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
8294 \space\glossentrydesc{##2}\glspostdescription\space ##3\par
8295 }%

```

Vertical gap between groups is the same as that used by indices:

```

8296 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}}

```

treegroup Like the tree style but the glossary groups have headings.

```

8297 \newglossarystyle{treegroup}{%

```

Base it on the glostyletree style:

```

8298 \setglossarystyle{tree}%

```



Each group has a heading (in bold) followed by a vertical gap):

```
8299 \renewcommand{\glsgroupheading}[1]{\par
8300 \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
8301 }
```

**treehypergroup** The treehypergroup style is like the treegroup style, but has a set of links to the groups at the start of the glossary.

```
8302 \newglossarystyle{treehypergroup}{%
```

Base it on the glostyletree style:

```
8303 \setglossarystyle{tree}%
```

Put navigation links to the groups at the start of the theglossary environment:

```
8304 \renewcommand*\glossaryheader}{%
8305 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
8306 \renewcommand*\glsgroupheading}[1]{%
8307 \par\noindent
8308 \textbf{\glsnavigationtarget{##1}{\glsgrouptitle{##1}}}\par
8309 \indexspace}%
8310 }
```

**\glstreeindent** Length governing left indent for each level of the tree style.

```
8311 \newlength\glstreeindent
8312 \setlength{\glstreeindent}{10pt}
```

**treenoname** The treenoname glossary style is like the tree style, but doesn't print the name or symbol for sub-levels.

```
8313 \newglossarystyle{treenoname}{%
```

Set the paragraph indentation and skip:

```
8314 \renewenvironment{theglossary}%
8315 {\setlength{\parindent}{0pt}%
8316 \setlength{\parskip}{0pt plus 0.3pt}}%
8317 {}%
```

No header:

```
8318 \renewcommand*\glossaryheader}{}%
```

No group headings:

```
8319 \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.

```
8320 \renewcommand{\glossentry}[2]{%
8321 \hangindent0pt\relax
8322 \parindent0pt\relax
8323 \glstentryitem{##1}\textbf{\glstarget{##1}{\glossentryname{##1}}}%
8324 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
8325 \space\glossentrydesc{##1}\glspostdescription\space##2\par
8326 }%
```

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.

```
8327 \renewcommand{\subglossentry}[3]{%
8328   \hangindent##1\glstreeindent\relax
8329   \parindent##1\glstreeindent\relax
8330   \ifnum##1=1\relax
8331     \glssubentryitem{##2}%
8332   \fi
8333   \glstarget{##2}{\strut}%
8334   \glossentrydesc{##2}\glspostdescription\space##3\par
8335 }
```

Vertical gap between groups is the same as that used by indices:

```
8336 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
8337 }
```

`treenonamegroup` Like the `treenoname` style but the glossary groups have headings.

```
8338 \newglossarystyle{treenonamegroup}{%
```

Base it on the `glostyletreenoname` style:

```
8339 \setglossarystyle{treenoname}%
```

Give each group a heading:

```
8340 \renewcommand{\glsgroupheading}[1]{\par
8341   \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
8342 }
```

`treenonamehypergroup` The `treenonamehypergroup` style is like the `treenonamegroup` style, but has a set of links to the groups at the start of the glossary.

```
8343 \newglossarystyle{treenonamehypergroup}{%
```

Base it on the `glostyletreenoname` style:

```
8344 \setglossarystyle{treenoname}%
```

Put navigation links to the groups at the start of the `theglossary` environment:

```
8345 \renewcommand*{\glossaryheader}{%
8346   \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
8347 \renewcommand*{\glsgroupheading}[1]{%
8348   \par\noindent
8349   \textbf{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}\par
8350   \indexspace}%
8351 }
```

`\glissetwidest` `\glissetwidest[ $\langle level \rangle$ ]{ $\langle text \rangle$ }` sets the widest text for the given level. It is used by the `alttree` glossary styles to determine the indentation of each level.

```
8352 \newcommand*{\glissetwidest}[2][0]{%
8353   \expandafter\def\csname @glswidestname\romannumeral#1\endcsname{%
8354     #2}%
8355 }
```

```

\@glswidestname Initialise \@glswidestname.
8356 \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.
8357 \newglossarystyle{almtree}{%
    Redefine theglossary environment.
8358 \renewenvironment{theglossary}%
8359     {\def\@gls@prevlevel{-1}%
8360      \mbox{}\par}%
8361     {\par}%

    Set the header and group headers to nothing.
8362 \renewcommand*{\glossaryheader}{}%
8363 \renewcommand*{\glsgroupheading}[1]{}%

    Redefine the way that the level 0 entries are displayed.
8364 \renewcommand{\glossentry}[2]{%
8365     \ifnum\@gls@prevlevel=0\relax
8366     \else

        Find out how big the indentation should be by measuring the widest entry.
8367         \settowidth{\glstreeindent}{\textbf{\@glswidestname\space}}%
8368         \fi

        Set the hangindent and paragraph indent.
8369         \hangindent\glstreeindent
8370         \parindent\glstreeindent

        Put the name to the left of the paragraph block.
8371         \makebox[0pt][r]{\makebox[\glstreeindent][l]{%
8372             \glsentryitem{##1}\textbf{\glstarget{##1}{\glossentryname{##1}}}}}%

        If the symbol is missing, ignore it, otherwise put it in brackets.
8373         \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%

        Do the description followed by the description terminator and location list.
8374         \glossentrydesc{##1}\glspostdescription \space ##2\par

        Set the previous level to 0.
8375         \def\@gls@prevlevel{0}%
8376     }%

    Redefine the way sub-entries are displayed.
8377 \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.
8378         \ifnum##1=1\relax
8379         \glssubentryitem{##2}%
8380         \fi

```

If the level hasn't changed, keep the same settings, otherwise adjust \glstreeindent accordingly.

```
8381 \ifnum\@gls@prevlevel=##1\relax
8382 \else
```

Compute the widest entry for this level, or for level 0 if not defined for this level.

Store in \gls@tmplen

```
8383 \@ifundefined{@glswidestname\romannumeral##1}{%
8384 \settowidth{\gls@tmplen}{\textbf{@glswidestname\space}}{%
8385 \settowidth{\gls@tmplen}{\textbf{%
8386 \csname @glswidestname\romannumeral##1\endcsname\space}}}%
```

Determine if going up or down a level

```
8387 \ifnum\@gls@prevlevel<##1\relax
```

Depth has increased, so add the width of the widest entry to \glstreeindent.

```
8388 \setlength\glstreeindent\gls@tmplen
8389 \addtolength\glstreeindent\parindent
8390 \parindent\glstreeindent
8391 \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.

```
8392 \@ifundefined{@glswidestname\romannumeral\@gls@prevlevel}{%
8393 \settowidth{\glstreeindent}{\textbf{%
8394 @glswidestname\space}}{%
8395 \settowidth{\glstreeindent}{\textbf{%
8396 \csname @glswidestname\romannumeral\@gls@prevlevel
8397 \endcsname\space}}}%
```

Subtract this length from the previous level's paragraph indent and set to \glstreeindent.

```
8398 \addtolength\parindent{-\glstreeindent}%
8399 \setlength\glstreeindent\parindent
8400 \fi
8401 \fi
```

Set the hanging indentation.

```
8402 \hangindent\glstreeindent
```

Put the name to the left of the paragraph block

```
8403 \makebox[0pt][r]{\makebox[\gls@tmplen][l]{%
8404 \textbf{\glstarget{##2}{\glossentryname{##2}}}}}%
```

If the symbol is missing, ignore it, otherwise put it in brackets.

```
8405 \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
```

Do the description followed by the description terminator and location list.

```
8406 \glossentrydesc{##2}\glspostdescription\space ##3\par
```

Set the previous level macro to the current level.

```
8407 \def\@gls@prevlevel{##1}%  
8408 }%
```

Vertical gap between groups is the same as that used by indices:

```
8409 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%  
8410 }
```

**alttreegroup** Like the `alttree` style but the glossary groups have headings.

```
8411 \newglossarystyle{alttreegroup}{%  
Base it on the glostylealttree style:  
8412 \setglossarystyle{alttree}%  
Give each group a heading.  
8413 \renewcommand{\glsgroupheading}[1]{\par  
8414 \def\@gls@prevlevel{-1}%  
8415 \hangindent0pt\relax  
8416 \parindent0pt\relax  
8417 \textbf{\glsgetgrouptitle{##1}}\par\indexspace}%  
8418 }
```

**alttreehypergroup** The `alttreehypergroup` style is like the `alttreegroup` style, but has a set of links to the groups at the start of the glossary.

```
8419 \newglossarystyle{alttreehypergroup}{%  
Base it on the glostylealttree style:  
8420 \setglossarystyle{alttree}%  
Put the navigation links in the header  
8421 \renewcommand*{\glossaryheader}{%  
8422 \par  
8423 \def\@gls@prevlevel{-1}%  
8424 \hangindent0pt\relax  
8425 \parindent0pt\relax  
8426 \textbf{\glsnavigation}\par\indexspace}%  
Put a hypertarget at the start of each group  
8427 \renewcommand*{\glsgroupheading}[1]{%  
8428 \par  
8429 \def\@gls@prevlevel{-1}%  
8430 \hangindent0pt\relax  
8431 \parindent0pt\relax  
8432 \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par  
8433 \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.

```

8434 \NeedsTeXFormat{LaTeX2e}
8435 \ProvidesPackage{glossaries-compatible-207}[2011/04/02 v1.0 (NLCT)]

```

`\GlsAddXdyAttribute` Adds an attribute in old format.

```

8436 \ifglxsindy
8437   \renewcommand*\GlsAddXdyAttribute[1]{%
8438     \edef\@xdyattributes{\@xdyattributes ^^J \string"#1\string"}%
8439     \expandafter\toks@\expandafter{\@xdylocref}%
8440     \edef\@xdylocref{\the\toks@ ^^J%
8441       (markup-locref
8442       :open \string"\string~\string\setentrycounter
8443       {\noexpand\glscounter}%
8444       \expandafter\string\csname#1\endcsname
8445       \expandafter\@gobble\string\{\string" ^^J
8446       :close \string"\expandafter\@gobble\string\}\string" ^^J
8447       :attr \string"#1\string"))}

```

Only has an effect before `\writeist`:

```

8448 \fi

```

`\GlsAddXdyCounters`

```

8449 \renewcommand*\GlsAddXdyCounters[1]{%
8450   \GlossariesWarning{\string\GlsAddXdyCounters\space not available
8451     in compatibility mode.}%
8452 }

```

Add predefined attributes

```

8453 \GlsAddXdyAttribute{glnumberformat}
8454 \GlsAddXdyAttribute{textrm}
8455 \GlsAddXdyAttribute{textsf}
8456 \GlsAddXdyAttribute{texttt}
8457 \GlsAddXdyAttribute{textbf}
8458 \GlsAddXdyAttribute{textmd}
8459 \GlsAddXdyAttribute{textit}
8460 \GlsAddXdyAttribute{textup}
8461 \GlsAddXdyAttribute{textsl}
8462 \GlsAddXdyAttribute{textsc}
8463 \GlsAddXdyAttribute{emph}
8464 \GlsAddXdyAttribute{glshypernumber}
8465 \GlsAddXdyAttribute{hyperrrm}
8466 \GlsAddXdyAttribute{hypersf}
8467 \GlsAddXdyAttribute{hypertt}
8468 \GlsAddXdyAttribute{hyperbf}
8469 \GlsAddXdyAttribute{hypermd}
8470 \GlsAddXdyAttribute{hyperit}
8471 \GlsAddXdyAttribute{hyperup}
8472 \GlsAddXdyAttribute{hypersl}
8473 \GlsAddXdyAttribute{hypersc}
8474 \GlsAddXdyAttribute{hyperemph}

```

\GlsAddXdyLocation Restore v2.07 definition:

```
8475 \ifglxsindy
8476   \renewcommand*{\GlsAddXdyLocation}[2]{%
8477     \edef\@xdyuserlocationdefs{%
8478       \@xdyuserlocationdefs ^^J%
8479       (define-location-class \string"#1\string"^^J\space\space
8480       \space(#2))
8481     }%
8482     \edef\@xdyuserlocationnames{%
8483       \@xdyuserlocationnames^^J\space\space\space
8484       \string"#1\string"}%
8485   }
8486 \fi
```

\@do@wrglossary

```
8487 \renewcommand{\@do@wrglossary}[1]{%
```

Determine whether to use xindy or makeindex syntax

```
8488 \ifglxsindy
```

Need to determine if the formatting information starts with a ( or ) indicating a range.

```
8489 \expandafter\@glo@check@mkidxrangechar\@glsnumberformat\@nil
8490 \def\@glo@range{}%
8491 \expandafter\if\@glo@prefix(\relax
8492   \def\@glo@range{:open-range}%
8493 \else
8494   \expandafter\if\@glo@prefix)\relax
8495   \def\@glo@range{:close-range}%
8496 \fi
8497 \fi
```

Get the location and escape any special characters

```
8498 \protected@edef\@glslocref{\theglsentrycounter}%
8499 \@gls@checkmkidxchars\@glslocref
```

Write to the glossary file using xindy syntax.

```
8500 \glossary[\csname glo@#1@type\endcsname]{%
8501 (indexentry :tkey (\csname glo@#1@index\endcsname)
8502   :locref \string"\@glslocref\string" %
8503   :attr \string"\@glo@suffix\string" \@glo@range
8504 )
8505 }%
8506 \else
```

Convert the format information into the format required for makeindex

```
8507 \@set@glo@numformat\@glo@numfmt\@gls@counter\@glsnumberformat
```

Write to the glossary file using makeindex syntax.

```
8508 \glossary[\csname glo@#1@type\endcsname]{%
8509 \string\glossaryentry{\csname glo@#1@index\endcsname
```

```

8510 \@gls@encapchar\@glo@numfmt}{\theglsentrycounter}}}%
8511 \fi
8512 }

```

\set@glo@numformat Only had 3 arguments in v2.07

```

8513 \def\set@glo@numformat#1#2#3{%
8514 \expandafter\@glo@check@mkidxrangechar#3\@nil
8515 \protected@edef#1{%
8516 \@glo@prefix setentrycounter[] {#2}%
8517 \expandafter\string\csname\@glo@suffix\endcsname
8518 }%
8519 \@gls@checkmkidxchars#1%
8520 }

```

\writeist Redefine \writeist back to the way it was in v2.07, but change \istfile to \glswrite.

```

8521 \ifglxsindy
8522 \def\writeist{%
8523 \openout\glswrite=\istfilename
8524 \write\glswrite{;; xindy style file created by the glossaries
8525 package in compatible-2.07 mode}%
8526 \write\glswrite{;; for document '\jobname' on
8527 \the\year-\the\month-\the\day}%
8528 \write\glswrite{^^J; required styles^^J}
8529 \@for\@xdystyle:=\@xdyrequiredstyles\do{%
8530 \ifx\@xdystyle\@empty
8531 \else
8532 \protected@write\glswrite{{(require
8533 \string"\@xdystyle.xdy\string")}}%
8534 \fi
8535 }%
8536 \write\glswrite{^^J%
8537 ; list of allowed attributes (number formats)^^J}%
8538 \write\glswrite{(define-attributes ((\@xdyattributes)))}%
8539 \write\glswrite{^^J; user defined alphabets^^J}%
8540 \write\glswrite{\@xdyuseralphabets}%
8541 \write\glswrite{^^J; location class definitions^^J}%
8542 \protected@edef\@gls@roman{\@roman{0}\string"
8543 \string"roman-numbers-lowercase\string" :sep \string"}}%
8544 \@onelevel@sanitize\@gls@roman
8545 \edef\@tmp{\string" \string"roman-numbers-lowercase\string"
8546 :sep \string"}%
8547 \@onelevel@sanitize\@tmp
8548 \ifx\@tmp\@gls@roman
8549 \write\glswrite{(define-location-class
8550 \string"roman-page-numbers\string"^^J\space\space\space
8551 (\string"roman-numbers-lowercase\string")
8552 :min-range-length \@glsminrange)}}%
8553 \else

```



```

8554 \write\glswrite{(define-location-class
8555 \string"roman-page-numbers\string"^^J\space\space\space
8556 (:sep "\@gls@roman")
8557 :min-range-length \@glsminrange)}}%
8558 \fi
8559 \write\glswrite{(define-location-class
8560 \string"Roman-page-numbers\string"^^J\space\space\space
8561 (\string"roman-numbers-uppercase\string")
8562 :min-range-length \@glsminrange)}}%
8563 \write\glswrite{(define-location-class
8564 \string"arabic-page-numbers\string"^^J\space\space\space
8565 (\string"arabic-numbers\string")
8566 :min-range-length \@glsminrange)}}%
8567 \write\glswrite{(define-location-class
8568 \string"alpha-page-numbers\string"^^J\space\space\space
8569 (\string"alpha\string")
8570 :min-range-length \@glsminrange)}}%
8571 \write\glswrite{(define-location-class
8572 \string"Alpha-page-numbers\string"^^J\space\space\space
8573 (\string"ALPHA\string")
8574 :min-range-length \@glsminrange)}}%
8575 \write\glswrite{(define-location-class
8576 \string"Appendix-page-numbers\string"^^J\space\space\space
8577 (\string"ALPHA\string"
8578 :sep \string"\@glsAlphacompositor\string"
8579 \string"arabic-numbers\string")
8580 :min-range-length \@glsminrange)}}%
8581 \write\glswrite{(define-location-class
8582 \string"arabic-section-numbers\string"^^J\space\space\space
8583 (\string"arabic-numbers\string"
8584 :sep \string"\glscompositor\string"
8585 \string"arabic-numbers\string")
8586 :min-range-length \@glsminrange)}}%
8587 \write\glswrite{^^J; user defined location classes}%
8588 \write\glswrite{\@xdyuserlocationdefs}%
8589 \write\glswrite{^^J; define cross-reference class^^J}%
8590 \write\glswrite{(define-crossref-class \string"see\string"
8591 :unverified )}%
8592 \write\glswrite{(markup-crossref-list
8593 :class \string"see\string"^^J\space\space\space
8594 :open \string"\string\glsseeformat\string"
8595 :close \string"{}\string")}%
8596 \write\glswrite{^^J; define the order of the location classes}%
8597 \write\glswrite{(define-location-class-order
8598 (\@xdylocationclassorder))}%
8599 \write\glswrite{^^J; define the glossary markup^^J}%
8600 \write\glswrite{(markup-index^^J\space\space\space
8601 :open \string"\string
8602 \glossarysection[\string\glossarytoctitle]{\string

```

```

8603     \glossarytitle}\string\glossarypreamble\string~n\string\begin
8604     {theglossary}\string\glossaryheader\string~n\string" ^^J\space
8605     \space\space:close \string"\expandafter\@gobble
8606     \string%\string~n\string
8607     \end{theglossary}\string\glossarypostamble
8608     \string~n\string" ^^J\space\space\space
8609     :tree)}}%
8610 \write\glswrite{(markup-letter-group-list
8611     :sep \string"\string\glsgroupskip\string~n\string"))}%
8612 \write\glswrite{(markup-indexentry
8613     :open \string"\string\relax \string\glsresetentrylist
8614     \string~n\string"))}%
8615 \write\glswrite{(markup-locclass-list :open
8616     \string"\glsopenbrace\string\glossaryentrynumbers
8617     \glsopenbrace\string\relax\space \string"^^J\space\space\space
8618     :sep \string", \string"
8619     :close \string"\glsclosebrace\glsclosebrace\string"))}%
8620 \write\glswrite{(markup-locref-list
8621     :sep \string"\string\delimN\space\string"))}%
8622 \write\glswrite{(markup-range
8623     :sep \string"\string\delimR\space\string"))}%
8624 \@onelevel@sanitize\gls@suffixF
8625 \@onelevel@sanitize\gls@suffixFF
8626 \ifx\gls@suffixF\@empty
8627 \else
8628     \write\glswrite{(markup-range
8629         :close "\gls@suffixF" :length 1 :ignore-end)}}%
8630 \fi
8631 \ifx\gls@suffixFF\@empty
8632 \else
8633     \write\glswrite{(markup-range
8634         :close "\gls@suffixFF" :length 2 :ignore-end)}}%
8635 \fi
8636 \write\glswrite{^^J; define format to use for locations^^J}%
8637 \write\glswrite{\@xdylocref}%
8638 \write\glswrite{^^J; define letter group list format^^J}%
8639 \write\glswrite{(markup-letter-group-list
8640     :sep \string"\string\glsgroupskip\string~n\string"))}%
8641 \write\glswrite{^^J; letter group headings^^J}%
8642 \write\glswrite{(markup-letter-group
8643     :open-head \string"\string\glsgroupheading
8644     \glsopenbrace\string"^^J\space\space\space
8645     :close-head \string"\glsclosebrace\string"))}%
8646 \write\glswrite{^^J; additional letter groups^^J}%
8647 \write\glswrite{\@xdylettergroups}%
8648 \write\glswrite{^^J; additional sort rules^^J}
8649 \write\glswrite{\@dysortrules}%
8650 \noist}
8651 \else

```

```

8652 \edef\@gls@actualchar{\string?}
8653 \edef\@gls@encapchar{\string|}
8654 \edef\@gls@levelchar{\string!}
8655 \edef\@gls@quotechar{\string"}
8656 \def\writeist{\relax
8657   \openout\glswrite=\istfilename
8658   \write\glswrite{\expandafter\@gobble\string\% makeindex style file
8659     created by the glossaries package}
8660   \write\glswrite{\expandafter\@gobble\string\% for document
8661     '\jobname' on \the\year-\the\month-\the\day}
8662   \write\glswrite{actual '\@gls@actualchar'}
8663   \write\glswrite{encap '\@gls@encapchar'}
8664   \write\glswrite{level '\@gls@levelchar'}
8665   \write\glswrite{quote '\@gls@quotechar'}
8666   \write\glswrite{keyword \string"\string\glossaryentry\string"}
8667   \write\glswrite{preamble \string"\string\glossarysection[\string
8668     \glossarytoctitle]{\string\glossarytitle}\string
8669     \glossarypreamble\string\n\string\begin{theglossary}\string
8670     \glossaryheader\string\n\string"}
8671   \write\glswrite{postamble \string"\string%\string\n\string
8672     \end{theglossary}\string\glossarypostamble\string\n
8673     \string"}
8674   \write\glswrite{group_skip \string"\string\glsgroupskip\string\n
8675     \string"}
8676   \write\glswrite{item_0 \string"\string%\string\n\string"}
8677   \write\glswrite{item_1 \string"\string%\string\n\string"}
8678   \write\glswrite{item_2 \string"\string%\string\n\string"}
8679   \write\glswrite{item_01 \string"\string%\string\n\string"}
8680   \write\glswrite{item_x1
8681     \string"\string\relax \string\glsresetentrylist\string\n
8682     \string"}
8683   \write\glswrite{item_12 \string"\string%\string\n\string"}
8684   \write\glswrite{item_x2
8685     \string"\string\relax \string\glsresetentrylist\string\n
8686     \string"}
8687   \write\glswrite{delim_0 \string"\string\{\string
8688     \glossaryentrynumbers\string\{\string\relax \string"}
8689   \write\glswrite{delim_1 \string"\string\{\string
8690     \glossaryentrynumbers\string\{\string\relax \string"}
8691   \write\glswrite{delim_2 \string"\string\{\string
8692     \glossaryentrynumbers\string\{\string\relax \string"}
8693   \write\glswrite{delim_t \string"\string\}\string\}\string"}
8694   \write\glswrite{delim_n \string"\string\delimN \string"}
8695   \write\glswrite{delim_r \string"\string\delimR \string"}
8696   \write\glswrite{headings_flag 1}
8697   \write\glswrite{heading_prefix
8698     \string"\string\glsgroupheading\string\{\string"}
8699   \write\glswrite{heading_suffix
8700     \string"\string\}\string\relax

```

```

8701     \string\glsresetentrylist \string"}
8702     \write\glswrite{symhead_positive \string"glssymbols\string"}
8703     \write\glswrite{numhead_positive \string"glsnnumbers\string"}
8704     \write\glswrite{page_compositor \string"\glscompositor\string"}
8705     \@gls@escbsdq\gls@suffixF
8706     \@gls@escbsdq\gls@suffixFF
8707     \ifx\gls@suffixF\@empty
8708     \else
8709         \write\glswrite{suffix_2p \string"\gls@suffixF\string"}
8710     \fi
8711     \ifx\gls@suffixFF\@empty
8712     \else
8713         \write\glswrite{suffix_3p \string"\gls@suffixFF\string"}
8714     \fi
8715     \noist
8716 }
8717 \fi

```

\noist

```

8718 \renewcommand*{\noist}{\let\writeist\relax}

```

Compatibility macros.

```

8719 \NeedsTeXFormat{LaTeX2e}
8720 \ProvidesPackage{glossaries-compatible-307}[2013/11/14 v4.0 (NLCT)]

```

Compatibility macros for predefined glossary styles:

**compatglossarystyle** Defines a compatibility glossary style.

```

8721 \newcommand{\compatglossarystyle}[2]{%
8722     \ifcsundef{@glscompstyle@#1}%
8723     {%
8724         \csdef{@glscompstyle@#1}{#2}%
8725     }%
8726     {%
8727         \PackageError{glossaries}{Glossary compatibility style ‘#1’ is already defined}{}%
8728     }%
8729 }

```

Backward compatible inline style.

```

8730 \compatglossarystyle{inline}{%
8731     \renewcommand{\glossaryentryfield}[5]{%
8732         \glsinlinedopostchild
8733         \gls@inlinesep
8734         \def\glo@desc{##3}%
8735         \def\@no@post@desc{\nopostdesc}%
8736         \glsentryitem{##1}\glsinlinenameformat{##1}{##2}%
8737         \ifx\glo@desc\@no@post@desc
8738             \glsinlineemptydescformat{##4}{##5}%
8739         \else
8740             \ifstrempy{##3}%

```

```

8741     {\glsinlineemptydescformat{##4}{##5}}%
8742     {\glsinlinedescformat{##3}{##4}{##5}}%
8743     \fi
8744     \ifglshaschildren{##1}%
8745     {%
8746         \glsresetsubentrycounter
8747         \glsinlineparentchildseparator
8748         \def\gls@inlinesubsep{%
8749             \def\gls@inlinepostchild{\glsinlinepostchild}%
8750         }%
8751     }%
8752     \def\gls@inlinesep{\glsinlineseparator}%
8753 }%

```

Sub-entries display description:

```

8754 \renewcommand{\glossarysubentryfield}[6]{%
8755     \gls@inlinesubsep%
8756     \glsinlinesubnameformat{##2}{##3}%
8757     \glssubentryitem{##2}\glsinlinesubdescformat{##4}{##5}{##6}%
8758     \def\gls@inlinesubsep{\glsinlinesubseparator}%
8759 }%
8760 }

```

Backward compatible list style.

```

8761 \compatglossarystyle{list}{%
8762     \renewcommand*\glossaryentryfield[5]{%
8763         \item[\glsentryitem{##1}\glstarget{##1}{##2}]
8764         ##3\glspostdescription\space ##5}%

```

Sub-entries continue on the same line:

```

8765 \renewcommand*\glossarysubentryfield[6]{%
8766     \glssubentryitem{##2}%
8767     \glstarget{##2}{\strut}##4\glspostdescription\space ##6.}%
8768 }

```

Backward compatible listgroup style.

```

8769 \compatglossarystyle{listgroup}{%
8770     \csuse{@glscompstyle@list}%
8771 }%

```

Backward compatible listhypergroup style.

```

8772 \compatglossarystyle{listhypergroup}{%
8773     \csuse{@glscompstyle@list}%
8774 }%

```

Backward compatible altlist style.

```

8775 \compatglossarystyle{altlist}{%
8776     \renewcommand*\glossaryentryfield[5]{%
8777         \item[\glsentryitem{##1}\glstarget{##1}{##2}]%
8778         \mbox{}\par\nobreak\@afterheading
8779         ##3\glspostdescription\space ##5}%
8780     \renewcommand{\glossarysubentryfield}[6]{%

```

```

8781 \par
8782 \glssubentryitem{##2}%
8783 \glstarget{##2}{\strut}##4\glspostdescription\space ##6}%
8784 }%

```

Backward compatible altlistgroup style.

```

8785 \compatglossarystyle{altlistgroup}{%
8786 \csuse{@glscmpstyle@altlist}%
8787 }%

```

Backward compatible altlisthypergroup style.

```

8788 \compatglossarystyle{altlisthypergroup}{%
8789 \csuse{@glscmpstyle@altlist}%
8790 }%

```

Backward compatible listdotted style.

```

8791 \compatglossarystyle{listdotted}{%
8792 \renewcommand*{\glossaryentryfield}[5]{%
8793 \item[]\makebox[\glslistdottedwidth][l]{%
8794 \glssubentryitem{##1}\glstarget{##1}{##2}%
8795 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##3}%
8796 \renewcommand*{\glossarysubentryfield}[6]{%
8797 \item[]\makebox[\glslistdottedwidth][l]{%
8798 \glssubentryitem{##2}%
8799 \glstarget{##2}{##3}%
8800 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##4}%
8801 }%

```

Backward compatible sublistdotted style.

```

8802 \compatglossarystyle{sublistdotted}{%
8803 \csuse{@glscmpstyle@listdotted}%
8804 \renewcommand*{\glossaryentryfield}[5]{%
8805 \item[\glssubentryitem{##1}\glstarget{##1}{##2}]}%
8806 }%

```

Backward compatible long style.

```

8807 \compatglossarystyle{long}{%
8808 \renewcommand*{\glossaryentryfield}[5]{%
8809 \glssubentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\}%
8810 \renewcommand*{\glossarysubentryfield}[6]{%
8811 &
8812 \glssubentryitem{##2}%
8813 \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\}%
8814 }%

```

Backward compatible longborder style.

```

8815 \compatglossarystyle{longborder}{%
8816 \csuse{@glscmpstyle@long}%
8817 }%

```

Backward compatible longheader style.

```

8818 \compatglossarystyle{longheader}{%

```

```
8819 \csuse{@glscompstyle@long}%
8820 }%
```

Backward compatible longheaderborder style.

```
8821 \compatglossarystyle{longheaderborder}{%
8822 \csuse{@glscompstyle@long}%
8823 }%
```

Backward compatible long3col style.

```
8824 \compatglossarystyle{long3col}{%
8825 \renewcommand*{\glossaryentryfield}[5]{%
8826 \glentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\\}%
8827 \renewcommand*{\glossarysubentryfield}[6]{%
8828 &
8829 \glsubentryitem{##2}%
8830 \glstarget{##2}{\strut}##4 & ##6\\}%
8831 }%
```

Backward compatible long3colborder style.

```
8832 \compatglossarystyle{long3colborder}{%
8833 \csuse{@glscompstyle@long3col}%
8834 }%
```

Backward compatible long3colheader style.

```
8835 \compatglossarystyle{long3colheader}{%
8836 \csuse{@glscompstyle@long3col}%
8837 }%
```

Backward compatible long3colheaderborder style.

```
8838 \compatglossarystyle{long3colheaderborder}{%
8839 \csuse{@glscompstyle@long3col}%
8840 }%
```

Backward compatible long4col style.

```
8841 \compatglossarystyle{long4col}{%
8842 \renewcommand*{\glossaryentryfield}[5]{%
8843 \glentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\\}%
8844 \renewcommand*{\glossarysubentryfield}[6]{%
8845 &
8846 \glsubentryitem{##2}%
8847 \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%
8848 }%
```

Backward compatible long4colheader style.

```
8849 \compatglossarystyle{long4colheader}{%
8850 \csuse{@glscompstyle@long4col}%
8851 }%
```

Backward compatible long4colborder style.

```
8852 \compatglossarystyle{long4colborder}{%
8853 \csuse{@glscompstyle@long4col}%
8854 }%
```

Backward compatible long4colheaderborder style.

```
8855 \compatglossarystyle{long4colheaderborder}{%  
8856 \csuse{@glscompstyle@long4col}%  
8857 }%
```

Backward compatible altlong4col style.

```
8858 \compatglossarystyle{altlong4col}{%  
8859 \csuse{@glscompstyle@long4col}%  
8860 }%
```

Backward compatible altlong4colheader style.

```
8861 \compatglossarystyle{altlong4colheader}{%  
8862 \csuse{@glscompstyle@long4col}%  
8863 }%
```

Backward compatible altlong4colborder style.

```
8864 \compatglossarystyle{altlong4colborder}{%  
8865 \csuse{@glscompstyle@long4col}%  
8866 }%
```

Backward compatible altlong4colheaderborder style.

```
8867 \compatglossarystyle{altlong4colheaderborder}{%  
8868 \csuse{@glscompstyle@long4col}%  
8869 }%
```

Backward compatible long style.

```
8870 \compatglossarystyle{longragged}{%  
8871 \renewcommand*{\glossaryentryfield}[5]{%  
8872 \glstarget{##1}{\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%  
8873 \tabularnewline}%  
8874 \renewcommand*{\glossarysubentryfield}[6]{%  
8875 &  
8876 \glssubentryitem{##2}%  
8877 \glstarget{##2}{\strut}##4\glspostdescription\space ##6%  
8878 \tabularnewline}%  
8879 }%
```

Backward compatible longraggedborder style.

```
8880 \compatglossarystyle{longraggedborder}{%  
8881 \csuse{@glscompstyle@longragged}%  
8882 }%
```

Backward compatible longraggedheader style.

```
8883 \compatglossarystyle{longraggedheader}{%  
8884 \csuse{@glscompstyle@longragged}%  
8885 }%
```

Backward compatible longraggedheaderborder style.

```
8886 \compatglossarystyle{longraggedheaderborder}{%  
8887 \csuse{@glscompstyle@longragged}%  
8888 }%
```



Backward compatible longragged3col style.

```
8889 \compatglossarystyle{longragged3col}{%
8890   \renewcommand*{\glossaryentryfield}[5]{%
8891     \glstarget{##1}{\glstarget{##1}{##2} & ##3 & ##5\tabularnewline}%
8892   \renewcommand*{\glossarysubentryfield}[6]{%
8893     &
8894     \glssubentryitem{##2}%
8895     \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%
8896 }%
```

Backward compatible longragged3colborder style.

```
8897 \compatglossarystyle{longragged3colborder}{%
8898   \csuse{@glscmpstyle@longragged3col}%
8899 }%
```

Backward compatible longragged3colheader style.

```
8900 \compatglossarystyle{longragged3colheader}{%
8901   \csuse{@glscmpstyle@longragged3col}%
8902 }%
```

Backward compatible longragged3colheaderborder style.

```
8903 \compatglossarystyle{longragged3colheaderborder}{%
8904   \csuse{@glscmpstyle@longragged3col}%
8905 }%
```

Backward compatible altlongragged4col style.

```
8906 \compatglossarystyle{altlongragged4col}{%
8907   \renewcommand*{\glossaryentryfield}[5]{%
8908     \glstarget{##1}{\glstarget{##1}{##2} & ##3 & ##4 & ##5\tabularnewline}%
8909   \renewcommand*{\glossarysubentryfield}[6]{%
8910     &
8911     \glssubentryitem{##2}%
8912     \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%
8913 }%
```

Backward compatible altlongragged4colheader style.

```
8914 \compatglossarystyle{altlongragged4colheader}{%
8915   \csuse{@glscmpstyle@altlong4col}%
8916 }%
```

Backward compatible altlongragged4colborder style.

```
8917 \compatglossarystyle{altlongragged4colborder}{%
8918   \csuse{@glscmpstyle@altlong4col}%
8919 }%
```

Backward compatible altlongragged4colheaderborder style.

```
8920 \compatglossarystyle{altlongragged4colheaderborder}{%
8921   \csuse{@glscmpstyle@altlong4col}%
8922 }%
```

Backward compatible index style.

```
8923 \compatglossarystyle{index}{%
```

```

8924 \renewcommand*\glossaryentryfield}[5]{%
8925 \item\glentryitem{##1}\textbf{\glstarget{##1}{##2}}%
8926 \ifx\relax##4\relax
8927 \else
8928 \space(##4)%
8929 \fi
8930 \space ##3\glspostdescription \space ##5}%
8931 \renewcommand*\glossarysubentryfield}[6]{%
8932 \ifcase##1\relax
8933 % level 0
8934 \item
8935 \or
8936 % level 1
8937 \subitem
8938 \glssubentryitem{##2}%
8939 \else
8940 % all other levels
8941 \subsubitem
8942 \fi
8943 \textbf{\glstarget{##2}{##3}}%
8944 \ifx\relax##5\relax
8945 \else
8946 \space(##5)%
8947 \fi
8948 \space##4\glspostdescription\space ##6}%
8949 }%

```

Backward compatible indexgroup style.

```

8950 \compatglossarystyle{indexgroup}{%
8951 \csuse{@glscmpstyle@index}%
8952 }%

```

Backward compatible indexhypergroup style.

```

8953 \compatglossarystyle{indexhypergroup}{%
8954 \csuse{@glscmpstyle@index}%
8955 }%

```

Backward compatible tree style.

```

8956 \compatglossarystyle{tree}{%
8957 \renewcommand*\glossaryentryfield}[5]{%
8958 \hangindent0pt\relax
8959 \parindent0pt\relax
8960 \glentryitem{##1}\textbf{\glstarget{##1}{##2}}%
8961 \ifx\relax##4\relax
8962 \else
8963 \space(##4)%
8964 \fi
8965 \space ##3\glspostdescription \space ##5\par}%
8966 \renewcommand*\glossarysubentryfield}[6]{%
8967 \hangindent##1\glstreeindent\relax
8968 \parindent##1\glstreeindent\relax

```

```

8969 \ifnum##1=1\relax
8970 \glssubentryitem{##2}%
8971 \fi
8972 \textbf{\glstarget{##2}{##3}}%
8973 \ifx\relax##5\relax
8974 \else
8975 \space(##5)%
8976 \fi
8977 \space##4\glspostdescription\space ##6\par}%
8978 }%

```

Backward compatible treegroup style.

```

8979 \compatglossarystyle{treegroup}{%
8980 \csuse{@glscmpstyle@tree}%
8981 }%

```

Backward compatible treehypergroup style.

```

8982 \compatglossarystyle{treehypergroup}{%
8983 \csuse{@glscmpstyle@tree}%
8984 }%

```

Backward compatible treenoname style.

```

8985 \compatglossarystyle{treenoname}{%
8986 \renewcommand{\glossaryentryfield}[5]{%
8987 \hangindent0pt\relax
8988 \parindent0pt\relax
8989 \glssubentryitem{##1}\textbf{\glstarget{##1}{##2}}%
8990 \ifx\relax##4\relax
8991 \else
8992 \space(##4)%
8993 \fi
8994 \space ##3\glspostdescription \space ##5\par}%
8995 \renewcommand{\glossarysubentryfield}[6]{%
8996 \hangindent##1\glstreeindent\relax
8997 \parindent##1\glstreeindent\relax
8998 \ifnum##1=1\relax
8999 \glssubentryitem{##2}%
9000 \fi
9001 \glstarget{##2}{\strut}%
9002 ##4\glspostdescription\space ##6\par}%
9003 }%

```

Backward compatible treenonamegroup style.

```

9004 \compatglossarystyle{treenonamegroup}{%
9005 \csuse{@glscmpstyle@treenoname}%
9006 }%

```

Backward compatible treenonamehypergroup style.

```

9007 \compatglossarystyle{treenonamehypergroup}{%
9008 \csuse{@glscmpstyle@treenoname}%
9009 }%

```

Backward compatible alttree style.

```

9010 \compatglossarystyle{alttree}{%
9011   \renewcommand{\glossaryentryfield}[5]{%
9012     \ifnum\@gls@prevlevel=0\relax
9013     \else
9014       \settowidth{\glstreeindent}{\textbf{\@glswidestname\space}}%
9015       \hangindent\glstreeindent
9016       \parindent\glstreeindent
9017     \fi
9018     \makebox[0pt][r]{\makebox[\glstreeindent][l]{%
9019       \glstryitem{##1}\textbf{\glstarget{##1}{##2}}}%
9020     \ifx\relax##4\relax
9021     \else
9022       (##4)\space
9023     \fi
9024     ##3\glspostdescription \space ##5\par
9025     \def\@gls@prevlevel{0}%
9026   }%
9027   \renewcommand{\glossarysubentryfield}[6]{%
9028     \ifnum##1=1\relax
9029       \glssubentryitem{##2}%
9030     \fi
9031     \ifnum\@gls@prevlevel=##1\relax
9032     \else
9033       \@ifundefined{\@glswidestname\romannumeral##1}{%
9034         \settowidth{\@gls@tmplen}{\textbf{\@glswidestname\space}}{%
9035         \settowidth{\@gls@tmplen}{\textbf{%
9036           \csname \@glswidestname\romannumeral##1\endcsname\space}}}%
9037       \ifnum\@gls@prevlevel<##1\relax
9038         \setlength\glstreeindent\@gls@tmplen
9039         \addtolength\glstreeindent\parindent
9040         \parindent\glstreeindent
9041       \else
9042         \@ifundefined{\@glswidestname\romannumeral\@gls@prevlevel}{%
9043           \settowidth{\glstreeindent}{\textbf{%
9044             \@glswidestname\space}}{%
9045           \settowidth{\glstreeindent}{\textbf{%
9046             \csname \@glswidestname\romannumeral\@gls@prevlevel
9047               \endcsname\space}}}%
9048           \addtolength\parindent{-\glstreeindent}%
9049           \setlength\glstreeindent\parindent
9050         \fi
9051       \fi
9052       \hangindent\glstreeindent
9053       \makebox[0pt][r]{\makebox[\@gls@tmplen][l]{%
9054         \textbf{\glstarget{##2}{##3}}}%
9055       \ifx##5\relax\relax
9056       \else
9057         (##5)\space

```

```

9058 \fi
9059 ##4\glspostdescription\space ##6\par
9060 \def\@gls@prevlevel{##1}%
9061 }%
9062 }%

```

Backward compatible alttreegroup style.

```

9063 \compatglossarystyle{alttreegroup}{%
9064 \csuse{@glscompstyle@almtree}%
9065 }%

```

Backward compatible alttreehypergroup style.

```

9066 \compatglossarystyle{alttreehypergroup}{%
9067 \csuse{@glscompstyle@almtree}%
9068 }%

```

Backward compatible mcolindex style.

```

9069 \compatglossarystyle{mcolindex}{%
9070 \csuse{@glscompstyle@index}%
9071 }%

```

Backward compatible mcolindexgroup style.

```

9072 \compatglossarystyle{mcolindexgroup}{%
9073 \csuse{@glscompstyle@index}%
9074 }%

```

Backward compatible mcolindexhypergroup style.

```

9075 \compatglossarystyle{mcolindexhypergroup}{%
9076 \csuse{@glscompstyle@index}%
9077 }%

```

Backward compatible mcoltree style.

```

9078 \compatglossarystyle{mcoltree}{%
9079 \csuse{@glscompstyle@tree}%
9080 }%

```

Backward compatible mcoltreegroup style.

```

9081 \compatglossarystyle{mcolindextreegroup}{%
9082 \csuse{@glscompstyle@tree}%
9083 }%

```

Backward compatible mcoltreehypergroup style.

```

9084 \compatglossarystyle{mcolindextreehypergroup}{%
9085 \csuse{@glscompstyle@tree}%
9086 }%

```

Backward compatible mcoltreenoname style.

```

9087 \compatglossarystyle{mcoltreenoname}{%
9088 \csuse{@glscompstyle@tree}%
9089 }%

```

Backward compatible mcoltreenonamegroup style.

```

9090 \compatglossarystyle{mcoltreenonamegroup}{%

```

9091 \csuse{@glscompstyle@tree}%  
 9092 }%

Backward compatible mcoltreenonamehypergroup style.

9093 \compatglossarystyle{mcoltreenonamehypergroup}{%  
 9094 \csuse{@glscompstyle@tree}%  
 9095 }%

Backward compatible mcolalmtree style.

9096 \compatglossarystyle{mcolalmtree}{%  
 9097 \csuse{@glscompstyle@almtree}%  
 9098 }%

Backward compatible mcolalmtreegroup style.

9099 \compatglossarystyle{mcolalmtreegroup}{%  
 9100 \csuse{@glscompstyle@almtree}%  
 9101 }%

Backward compatible mcolalmtreehypergroup style.

9102 \compatglossarystyle{mcolalmtreehypergroup}{%  
 9103 \csuse{@glscompstyle@almtree}%  
 9104 }%

Backward compatible superragged style.

9105 \compatglossarystyle{superragged}{%  
 9106 \renewcommand\*{\glossaryentryfield}[5]{%  
 9107 \glsentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%  
 9108 \tabularnewline}%  
 9109 \renewcommand\*{\glossarysubentryfield}[6]{%  
 9110 &  
 9111 \glssubentryitem{##2}%  
 9112 \glstarget{##2}{\strut}##4\glspostdescription\space ##6%  
 9113 \tabularnewline}%  
 9114 }%

Backward compatible superraggedborder style.

9115 \compatglossarystyle{superraggedborder}{%  
 9116 \csuse{@glscompstyle@superragged}%  
 9117 }%

Backward compatible superraggedheader style.

9118 \compatglossarystyle{superraggedheader}{%  
 9119 \csuse{@glscompstyle@superragged}%  
 9120 }%

Backward compatible superraggedheaderborder style.

9121 \compatglossarystyle{superraggedheaderborder}{%  
 9122 \csuse{@glscompstyle@superragged}%  
 9123 }%

Backward compatible superragged3col style.

9124 \compatglossarystyle{superragged3col}{%  
 9125 \renewcommand\*{\glossaryentryfield}[5]{%

```

9126 \glentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\tabularnewline}%
9127 \renewcommand*{\glossarysubentryfield}[6]{%
9128     &
9129     \glssubentryitem{##2}%
9130     \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%
9131 }%

```

Backward compatible superragged3colborder style.

```

9132 \compatglossarystyle{superragged3colborder}{%
9133 \csuse{@glscmpstyle@superragged3col}%
9134 }%

```

Backward compatible superragged3colheader style.

```

9135 \compatglossarystyle{superragged3colheader}{%
9136 \csuse{@glscmpstyle@superragged3col}%
9137 }%

```

Backward compatible superragged3colheaderborder style.

```

9138 \compatglossarystyle{superragged3colheaderborder}{%
9139 \csuse{@glscmpstyle@superragged3col}%
9140 }%

```

Backward compatible altsuperragged4col style.

```

9141 \compatglossarystyle{altsuperragged4col}{%
9142 \renewcommand*{\glossaryentryfield}[5]{%
9143     \glentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\tabularnewline}%
9144 \renewcommand*{\glossarysubentryfield}[6]{%
9145     &
9146     \glssubentryitem{##2}%
9147     \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%
9148 }%

```

Backward compatible altsuperragged4colheader style.

```

9149 \compatglossarystyle{altsuperragged4colheader}{%
9150 \csuse{@glscmpstyle@altsuperragged4col}%
9151 }%

```

Backward compatible altsuperragged4colborder style.

```

9152 \compatglossarystyle{altsuperragged4colborder}{%
9153 \csuse{@glscmpstyle@altsuperragged4col}%
9154 }%

```

Backward compatible altsuperragged4colheaderborder style.

```

9155 \compatglossarystyle{altsuperragged4colheaderborder}{%
9156 \csuse{@glscmpstyle@altsuperragged4col}%
9157 }%

```

Backward compatible super style.

```

9158 \compatglossarystyle{super}{%
9159 \renewcommand*{\glossaryentryfield}[5]{%
9160     \glentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\}%
9161 \renewcommand*{\glossarysubentryfield}[6]{%

```

```

9162      &
9163      \glssubentryitem{##2}%
9164      \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\}%
9165 }%

```

Backward compatible superborder style.

```

9166 \compatglossarystyle{superborder}{%
9167 \csuse{@glscmpstyle@super}%
9168 }%

```

Backward compatible superheader style.

```

9169 \compatglossarystyle{superheader}{%
9170 \csuse{@glscmpstyle@super}%
9171 }%

```

Backward compatible superheaderborder style.

```

9172 \compatglossarystyle{superheaderborder}{%
9173 \csuse{@glscmpstyle@super}%
9174 }%

```

Backward compatible super3col style.

```

9175 \compatglossarystyle{super3col}{%
9176 \renewcommand*{\glossaryentryfield}[5]{%
9177 \glssentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\\}%
9178 \renewcommand*{\glossarysubentryfield}[6]{%
9179 &
9180 \glssubentryitem{##2}%
9181 \glstarget{##2}{\strut}##4 & ##6\\}%
9182 }%

```

Backward compatible super3colborder style.

```

9183 \compatglossarystyle{super3colborder}{%
9184 \csuse{@glscmpstyle@super3col}%
9185 }%

```

Backward compatible super3colheader style.

```

9186 \compatglossarystyle{super3colheader}{%
9187 \csuse{@glscmpstyle@super3col}%
9188 }%

```

Backward compatible super3colheaderborder style.

```

9189 \compatglossarystyle{super3colheaderborder}{%
9190 \csuse{@glscmpstyle@super3col}%
9191 }%

```

Backward compatible super4col style.

```

9192 \compatglossarystyle{super4col}{%
9193 \renewcommand*{\glossaryentryfield}[5]{%
9194 \glssentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\\}%
9195 \renewcommand*{\glossarysubentryfield}[6]{%
9196 &
9197 \glssubentryitem{##2}%

```



```

9198 \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%
9199 }%

```

Backward compatible super4colheader style.

```

9200 \compatglossarystyle{super4colheader}{%
9201 \csuse{@glscmpstyle@super4col}%
9202 }%

```

Backward compatible super4colborder style.

```

9203 \compatglossarystyle{super4colborder}{%
9204 \csuse{@glscmpstyle@super4col}%
9205 }%

```

Backward compatible super4colheaderborder style.

```

9206 \compatglossarystyle{super4colheaderborder}{%
9207 \csuse{@glscmpstyle@super4col}%
9208 }%

```

Backward compatible altsuper4col style.

```

9209 \compatglossarystyle{altsuper4col}{%
9210 \csuse{@glscmpstyle@super4col}%
9211 }%

```

Backward compatible altsuper4colheader style.

```

9212 \compatglossarystyle{altsuper4colheader}{%
9213 \csuse{@glscmpstyle@super4col}%
9214 }%

```

Backward compatible altsuper4colborder style.

```

9215 \compatglossarystyle{altsuper4colborder}{%
9216 \csuse{@glscmpstyle@super4col}%
9217 }%

```

Backward compatible altsuper4colheaderborder style.

```

9218 \compatglossarystyle{altsuper4colheaderborder}{%
9219 \csuse{@glscmpstyle@super4col}%
9220 }%

```

## 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.

```

9221 \NeedsTeXFormat{LaTeX2e}

```

Package version number now in line with main glossaries package number but will only be updated when glossaries-accsupp.sty is modified.

```

9222 \ProvidesPackage{glossaries-accsupp}[2014/03/06 v4.04 (NLCT)
9223 Experimental glossaries accessibility]

```

Pass all options to glossaries:

```

9224 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{glossaries}}

```

Process options:

```
9225 \ProcessOptions
```

ompatibleglossentry   Override style compatibility macros:

```
9226 \def\compatibleglossentry#1#2{%
9227   \toks@{#2}%
9228   \protected@edef\@do@glossentry{%
9229     \noexpand\accsuppglossaryentryfield{#1}%
9230     {\noexpand\glsnamefont
9231       {\expandafter\expandonce\csname glo@\glsdetoklabel{#1}@name\endcsname}}}%
9232     {\expandafter\expandonce\csname glo@\glsdetoklabel{#1}@desc\endcsname}%
9233     {\expandafter\expandonce\csname glo@\glsdetoklabel{#1}@symbol\endcsname}%
9234     {\the\toks@}%
9235   }%
9236   \@do@glossentry
9237 }
```

atiblesubglossentry

```
9238 \def\compatiblesubglossentry#1#2#3{%
9239   \toks@{#3}%
9240   \protected@edef\@do@subglossentry{%
9241     \noexpand\accsuppglossarysubentryfield{\number#1}%
9242     {#2}%
9243     {\noexpand\glsnamefont
9244       {\expandafter\expandonce\csname glo@\glsdetoklabel{#2}@name\endcsname}}}%
9245     {\expandafter\expandonce\csname glo@\glsdetoklabel{#2}@desc\endcsname}%
9246     {\expandafter\expandonce\csname glo@\glsdetoklabel{#2}@symbol\endcsname}%
9247     {\the\toks@}%
9248   }%
9249   \@do@subglossentry
9250 }
```

Required packages:

```
9251 \RequirePackage{glossaries}
9252 \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:

```
9253 \define@key{glossentry}{access}{%
9254   \def\@glo@access{#1}%
9255 }
```

textaccess The replacement text corresponding to the text key:

```
9256 \define@key{glossentry}{textaccess}{%
9257   \def\@glo@textaccess{#1}%
9258 }
```

firstaccess The replacement text corresponding to the first key:

```
9259 \define@key{glossentry}{firstaccess}{%
9260   \def\@glo@firstaccess{#1}%
9261 }
```

pluralaccess The replacement text corresponding to the plural key:

```
9262 \define@key{glossentry}{pluralaccess}{%
9263   \def\@glo@pluralaccess{#1}%
9264 }
```

firstpluralaccess The replacement text corresponding to the firstplural key:

```
9265 \define@key{glossentry}{firstpluralaccess}{%
9266   \def\@glo@firstpluralaccess{#1}%
9267 }
```

symbolaccess The replacement text corresponding to the symbol key:

```
9268 \define@key{glossentry}{symbolaccess}{%
9269   \def\@glo@symbolaccess{#1}%
9270 }
```

symbolpluralaccess The replacement text corresponding to the symbolplural key:

```
9271 \define@key{glossentry}{symbolpluralaccess}{%
9272   \def\@glo@symbolpluralaccess{#1}%
9273 }
```

descriptionaccess The replacement text corresponding to the description key:

```
9274 \define@key{glossentry}{descriptionaccess}{%
9275   \def\@glo@descaccess{#1}%
9276 }
```

descriptionpluralaccess The replacement text corresponding to the descriptionplural key:

```
9277 \define@key{glossentry}{descriptionpluralaccess}{%
9278   \def\@glo@descpluralaccess{#1}%
9279 }
```

shortaccess The replacement text corresponding to the short key:

```
9280 \define@key{glossentry}{shortaccess}{%
9281   \def\@glo@shortaccess{#1}%
9282 }
```

shortpluralaccess The replacement text corresponding to the shortplural key:

```
9283 \define@key{glossentry}{shortpluralaccess}{%
9284   \def\@glo@shortpluralaccess{#1}%
9285 }
```

longaccess The replacement text corresponding to the long key:

```
9286 \define@key{glossentry}{longaccess}{%
9287   \def\@glo@longaccess{#1}%
9288 }
```

longpluralaccess The replacement text corresponding to the longplural key:

```
9289 \define@key{glossentry}{longpluralaccess}{%
9290   \def\@glo@longpluralaccess{#1}%
9291 }
```

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:

```
9292 \appto\@gls@keymap{,%
9293   {access}{access},%
9294   {textaccess}{textaccess},%
9295   {firstaccess}{firstaccess},%
9296   {pluralaccess}{pluralaccess},%
9297   {firstpluralaccess}{firstpluralaccess},%
9298   {symbolaccess}{symbolaccess},%
9299   {symbolpluralaccess}{symbolpluralaccess},%
9300   {descaccess}{descaccess},%
9301   {descpluralaccess}{descpluralaccess},%
9302   {shortaccess}{shortaccess},%
9303   {shortpluralaccess}{shortpluralaccess},%
9304   {longaccess}{longaccess},%
9305   {longpluralaccess}{longpluralaccess}%
9306 }
```

\@gls@noaccess Indicates that no replacement text has been provided.

```
9307 \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):

```
9308 \let\@gls@oldnewglossaryentryprehook\@newglossaryentryprehook
9309 \renewcommand*{\@newglossaryentryprehook}{%
9310   \@gls@oldnewglossaryentryprehook
9311   \def\@glo@access{\@glo@symbol}%
9312 }
```

Initialise the other keys:

```
9312 \def\@glo@textaccess{\@glo@access}%
9313 \def\@glo@firstaccess{\@glo@access}%
9314 \def\@glo@pluralaccess{\@glo@textaccess}%
9315 \def\@glo@firstpluralaccess{\@glo@pluralaccess}%
9316 \def\@glo@symbolaccess{\relax}%
9317 \def\@glo@symbolpluralaccess{\@glo@symbolaccess}%
9318 \def\@glo@descaccess{\relax}%
9319 \def\@glo@descpluralaccess{\@glo@descaccess}%
9320 }
```

```

9320 \def\@glo@shortaccess{\relax}%
9321 \def\@glo@shortpluralaccess{\@glo@shortaccess}%
9322 \def\@glo@longaccess{\relax}%
9323 \def\@glo@longpluralaccess{\@glo@longaccess}%
9324 }

```

Add to the end hook:

```

9325 \let\@gls@oldnewglossaryentryposthook\@newglossaryentryposthook
9326 \renewcommand*{\@newglossaryentryposthook}{%
9327 \@gls@oldnewglossaryentryposthook

```

Store the access information:

```

9328 \expandafter
9329 \protected@xdef\csname glo@\@glo@label @access\endcsname{%
9330 \@glo@access}%
9331 \expandafter
9332 \protected@xdef\csname glo@\@glo@label @textaccess\endcsname{%
9333 \@glo@textaccess}%
9334 \expandafter
9335 \protected@xdef\csname glo@\@glo@label @firstaccess\endcsname{%
9336 \@glo@firstaccess}%
9337 \expandafter
9338 \protected@xdef\csname glo@\@glo@label @pluralaccess\endcsname{%
9339 \@glo@pluralaccess}%
9340 \expandafter
9341 \protected@xdef\csname glo@\@glo@label @firstpluralaccess\endcsname{%
9342 \@glo@firstpluralaccess}%
9343 \expandafter
9344 \protected@xdef\csname glo@\@glo@label @symbolaccess\endcsname{%
9345 \@glo@symbolaccess}%
9346 \expandafter
9347 \protected@xdef\csname glo@\@glo@label @symbolpluralaccess\endcsname{%
9348 \@glo@symbolpluralaccess}%
9349 \expandafter
9350 \protected@xdef\csname glo@\@glo@label @descaccess\endcsname{%
9351 \@glo@descaccess}%
9352 \expandafter
9353 \protected@xdef\csname glo@\@glo@label @descpluralaccess\endcsname{%
9354 \@glo@descpluralaccess}%
9355 \expandafter
9356 \protected@xdef\csname glo@\@glo@label @shortaccess\endcsname{%
9357 \@glo@shortaccess}%
9358 \expandafter
9359 \protected@xdef\csname glo@\@glo@label @shortpluralaccess\endcsname{%
9360 \@glo@shortpluralaccess}%
9361 \expandafter
9362 \protected@xdef\csname glo@\@glo@label @longaccess\endcsname{%
9363 \@glo@longaccess}%
9364 \expandafter
9365 \protected@xdef\csname glo@\@glo@label @longpluralaccess\endcsname{%

```

```

9366      \@glo@longpluralaccess}%
9367 }

```

## 6.2 Accessing Replacement Text

`\glsentryaccess` Get the value of the access key for the entry with the given label:

```

9368 \newcommand*{\glsentryaccess}[1]{%
9369   \@gls@entry@field{#1}{access}%
9370 }

```

`\glsentrytextaccess` Get the value of the textaccess key for the entry with the given label:

```

9371 \newcommand*{\glsentrytextaccess}[1]{%
9372   \@gls@entry@field{#1}{textaccess}%
9373 }

```

`\glsentryfirstaccess` Get the value of the firstaccess key for the entry with the given label:

```

9374 \newcommand*{\glsentryfirstaccess}[1]{%
9375   \@gls@entry@field{#1}{firstaccess}%
9376 }

```

`\glsentrypluralaccess` Get the value of the pluralaccess key for the entry with the given label:

```

9377 \newcommand*{\glsentrypluralaccess}[1]{%
9378   \@gls@entry@field{#1}{pluralaccess}%
9379 }

```

`\glsentryfirstpluralaccess` Get the value of the firstpluralaccess key for the entry with the given label:

```

9380 \newcommand*{\glsentryfirstpluralaccess}[1]{%
9381   \csname glo@#1@firstpluralaccess\endcsname
9382 }

```

`\glsentrysymbolaccess` Get the value of the symbolaccess key for the entry with the given label:

```

9383 \newcommand*{\glsentrysymbolaccess}[1]{%
9384   \@gls@entry@field{#1}{symbolaccess}%
9385 }

```

`\glsentrysymbolpluralaccess` Get the value of the symbolpluralaccess key for the entry with the given label:

```

9386 \newcommand*{\glsentrysymbolpluralaccess}[1]{%
9387   \@gls@entry@field{#1}{symbolpluralaccess}%
9388 }

```

`\glsentrydescaccess` Get the value of the descriptionaccess key for the entry with the given label:

```

9389 \newcommand*{\glsentrydescaccess}[1]{%
9390   \@gls@entry@field{#1}{descaccess}%
9391 }

```

`\glsentrydescpluralaccess` Get the value of the descriptionpluralaccess key for the entry with the given label:

```

9392 \newcommand*{\glsentrydescpluralaccess}[1]{%
9393   \@gls@entry@field{#1}{descaccess}%
9394 }

```

`\glsentryshortaccess` Get the value of the shortaccess key for the entry with the given label:

```

9395 \newcommand*{\glsentryshortaccess}[1]{%
9396   \@gls@entry@field{#1}{shortaccess}%
9397 }

```

`\glsentryshortpluralaccess` Get the value of the shortpluralaccess key for the entry with the given label:

```

9398 \newcommand*{\glsentryshortpluralaccess}[1]{%
9399   \@gls@entry@field{#1}{shortpluralaccess}%
9400 }

```

`\glsentrylongaccess` Get the value of the longaccess key for the entry with the given label:

```

9401 \newcommand*{\glsentrylongaccess}[1]{%
9402   \@gls@entry@field{#1}{longaccess}%
9403 }

```

`\glsentrylongpluralaccess` Get the value of the longpluralaccess key for the entry with the given label:

```

9404 \newcommand*{\glsentrylongpluralaccess}[1]{%
9405   \@gls@entry@field{#1}{longpluralaccess}%
9406 }

```

`\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.)

```

9407 \newcommand*{\glsaccsupp}[2]{%
9408   \BeginAccSupp{ActualText=#1}#2\EndAccSupp{}%
9409 }

```

`\xglsaccsupp` Fully expands replacement text before calling `\glsaccsupp`

```

9410 \newcommand*{\xglsaccsupp}[2]{%
9411   \protected@edef\@gls@replacementtext{#1}%
9412   \expandafter\glsaccsupp\expandafter{\@gls@replacementtext}{#2}%
9413 }

```

`\@gls@access@display`

```

9414 \newcommand*{\@gls@access@display}[2]{%
9415   \protected@edef\@glo@access{#2}%
9416   \ifx\@glo@access\@gls@noaccess
9417     #1%
9418   \else
9419     \xglsaccsupp{\@glo@access}{#1}%
9420   \fi
9421 }

```

l<sub>sname</sub>accessdisplay    Displays the first argument with the accessibility text for the entry with the label given by the second argument (if set).

```
9422 \DeclareRobustCommand*{\glsnameaccessdisplay}[2]{%
9423   \@gls@access@display{#1}{\glsentryaccess{#2}}}%
9424 }
```

l<sub>stext</sub>accessdisplay    As above but for the text<sub>access</sub> replacement text.

```
9425 \DeclareRobustCommand*{\glstextaccessdisplay}[2]{%
9426   \@gls@access@display{#1}{\glsentrytextaccess{#2}}}%
9427 }
```

pl<sub>ural</sub>accessdisplay    As above but for the plural<sub>access</sub> replacement text.

```
9428 \DeclareRobustCommand*{\glspluralaccessdisplay}[2]{%
9429   \@gls@access@display{#1}{\glsentrypluralaccess{#2}}}%
9430 }
```

s<sub>first</sub>accessdisplay    As above but for the first<sub>access</sub> replacement text.

```
9431 \DeclareRobustCommand*{\glsfirstaccessdisplay}[2]{%
9432   \@gls@access@display{#1}{\glsentryfirstaccess{#2}}}%
9433 }
```

pl<sub>ural</sub>accessdisplay    As above but for the first<sub>plural</sub>access replacement text.

```
9434 \DeclareRobustCommand*{\glsfirstpluralaccessdisplay}[2]{%
9435   \@gls@access@display{#1}{\glsentryfirstpluralaccess{#2}}}%
9436 }
```

s<sub>ymbol</sub>accessdisplay    As above but for the symbol<sub>access</sub> replacement text.

```
9437 \DeclareRobustCommand*{\glssymbolaccessdisplay}[2]{%
9438   \@gls@access@display{#1}{\glsentrysymbolaccess{#2}}}%
9439 }
```

pl<sub>ural</sub>accessdisplay    As above but for the symbol<sub>plural</sub>access replacement text.

```
9440 \DeclareRobustCommand*{\glssymbolpluralaccessdisplay}[2]{%
9441   \@gls@access@display{#1}{\glsentrysymbolpluralaccess{#2}}}%
9442 }
```

i<sub>ption</sub>accessdisplay    As above but for the description<sub>access</sub> replacement text.

```
9443 \DeclareRobustCommand*{\glsdescriptionaccessdisplay}[2]{%
9444   \@gls@access@display{#1}{\glsentrydescaccess{#2}}}%
9445 }
```

pl<sub>ural</sub>accessdisplay    As above but for the description<sub>plural</sub>access replacement text.

```
9446 \DeclareRobustCommand*{\glsdescriptionpluralaccessdisplay}[2]{%
9447   \@gls@access@display{#1}{\glsentrydescpluralaccess{#2}}}%
9448 }
```



sshortaccessdisplay As above but for the shortaccess replacement text.

```

9449 \DeclareRobustCommand*\glsshortaccessdisplay}[2]{%
9450   \@gls@access@display{#1}{\glsentryshortaccess{#2}}%
9451 }

```

pluralaccessdisplay As above but for the shortpluralaccess replacement text.

```

9452 \DeclareRobustCommand*\glsshortpluralaccessdisplay}[2]{%
9453   \@gls@access@display{#1}{\glsentryshortpluralaccess{#2}}%
9454 }

```

lslongaccessdisplay As above but for the longaccess replacement text.

```

9455 \DeclareRobustCommand*\glslongaccessdisplay}[2]{%
9456   \@gls@access@display{#1}{\glsentrylongaccess{#2}}%
9457 }

```

pluralaccessdisplay As above but for the longpluralaccess replacement text.

```

9458 \DeclareRobustCommand*\glslongpluralaccessdisplay}[2]{%
9459   \@gls@access@display{#1}{\glsentrylongpluralaccess{#2}}%
9460 }

```

\glsaccessdisplay Gets the replacement text corresponding to the named key given by the first argument and calls the appropriate command defined above.

```

9461 \DeclareRobustCommand*\glsaccessdisplay}[3]{%
9462   \@ifundefined{gls#1accessdisplay}%
9463   {%
9464     \PackageError{glossaries-accsupp}{No accessibility support
9465       for key ‘#1’}{}%
9466   }%
9467   {%
9468     \csname gls#1accessdisplay\endcsname{#2}{#3}%
9469   }%
9470 }

```

lsls@default@entryfmt Redefine the default entry format to use accessibility information

```

9471 \renewcommand*\@@gls@default@entryfmt}[2]{%
9472   \ifdefempty\glscustomtext
9473   {%
9474     \glsifplural
9475     {%

```

Plural form

```

9476       \glscapscase
9477       {%

```

Don't adjust case

```

9478       \ifglsused\glslabel
9479       {%

```

Subsequent use

```

9480      #2{\glspluralaccessdisplay
9481          {\glsentryplural{\glslabel}}{\glslabel}}%
9482          {\glsdescriptionpluralaccessdisplay
9483              {\glsentrydescplural{\glslabel}}{\glslabel}}%
9484          {\glssymbolpluralaccessdisplay
9485              {\glsentrysymbolplural{\glslabel}}{\glslabel}}
9486          {\glsinsert}}%
9487      }%
9488      {%

```

First use

```

9489      #1{\glsfirstpluralaccessdisplay
9490          {\glsentryfirstplural{\glslabel}}{\glslabel}}%
9491          {\glsdescriptionpluralaccessdisplay
9492              {\glsentrydescplural{\glslabel}}{\glslabel}}%
9493          {\glssymbolpluralaccessdisplay
9494              {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
9495          {\glsinsert}}%
9496      }%
9497      }%
9498      {%

```

Make first letter upper case

```

9499      \ifglsused\glslabel
9500      {%

```

Subsequent use.

```

9501      #2{\glspluralaccessdisplay
9502          {\Glsentryplural{\glslabel}}{\glslabel}}%
9503          {\glsdescriptionpluralaccessdisplay
9504              {\glsentrydescplural{\glslabel}}{\glslabel}}%
9505          {\glssymbolpluralaccessdisplay
9506              {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
9507          {\glsinsert}}%
9508      }%
9509      {%

```

First use

```

9510      #1{\glsfirstpluralaccessdisplay
9511          {\Glsentryfirstplural{\glslabel}}{\glslabel}}%
9512          {\glsdescriptionpluralaccessdisplay
9513              {\glsentrydescplural{\glslabel}}{\glslabel}}%
9514          {\glssymbolpluralaccessdisplay
9515              {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
9516          {\glsinsert}}%
9517      }%
9518      }%
9519      {%

```

Make all upper case

```

9520      \ifglused\glslabel
9521      {%

```

#### Subsequent use

```

9522      \MakeUppercase{%
9523      #2{\glspluralaccessdisplay
9524      {\glentryplural{\glslabel}}{\glslabel}}%
9525      {\gldescriptionpluralaccessdisplay
9526      {\glentrydescplural{\glslabel}}{\glslabel}}%
9527      {\glssymbolpluralaccessdisplay
9528      {\glentrysymbolplural{\glslabel}}{\glslabel}}%
9529      {\glinsert}}%
9530      }%
9531      {%

```

#### First use

```

9532      \MakeUppercase{%
9533      #1{\glfirstpluralaccessdisplay
9534      {\glentryfirstplural{\glslabel}}{\glslabel}}%
9535      {\gldescriptionpluralaccessdisplay
9536      {\glentrydescplural{\glslabel}}{\glslabel}}%
9537      {\glssymbolpluralaccessdisplay
9538      {\glentrysymbolplural{\glslabel}}{\glslabel}}%
9539      {\glinsert}}%
9540      }%
9541      }%
9542      }%
9543      {%

```

#### Singular form

```

9544      \glscapscase
9545      {%

```

#### Don't adjust case

```

9546      \ifglused\glslabel
9547      {%

```

#### Subsequent use

```

9548      #2{\glstextaccessdisplay
9549      {\glentrytext{\glslabel}}{\glslabel}}%
9550      {\gldescriptionaccessdisplay
9551      {\glentrydesc{\glslabel}}{\glslabel}}%
9552      {\glssymbolaccessdisplay
9553      {\glentrysymbol{\glslabel}}{\glslabel}}%
9554      {\glinsert}}%
9555      }%
9556      {%

```

#### First use

```

9557      #1{\glfirstaccessdisplay
9558      {\glentryfirst{\glslabel}}{\glslabel}}%
9559      {\gldescriptionaccessdisplay

```

```

9560         {\glentrydesc{\glslabel}}{\glslabel}}%
9561     {\glssymbolaccessdisplay
9562         {\glentrysymbol{\glslabel}}{\glslabel}}%
9563     {\glsinsert}%
9564 }%
9565 }%
9566 {%

```

Make first letter upper case

```

9567     \ifglsused\glslabel
9568     {%

```

Subsequent use

```

9569     #2{\glstextaccessdisplay
9570         {\Glsentrytext{\glslabel}}{\glslabel}}%
9571     {\glsdescriptionaccessdisplay
9572         {\glentrydesc{\glslabel}}{\glslabel}}%
9573     {\glssymbolaccessdisplay
9574         {\glentrysymbol{\glslabel}}{\glslabel}}%
9575     {\glsinsert}%
9576 }%
9577 {%

```

First use

```

9578     #1{\glsfirstaccessdisplay
9579         {\Glsentryfirst{\glslabel}}{\glslabel}}%
9580     {\glsdescriptionaccessdisplay
9581         {\glentrydesc{\glslabel}}{\glslabel}}%
9582     {\glssymbolaccessdisplay
9583         {\glentrysymbol{\glslabel}}{\glslabel}}%
9584     {\glsinsert}%
9585 }%
9586 }%
9587 {%

```

Make all upper case

```

9588     \ifglsused\glslabel
9589     {%

```

Subsequent use

```

9590     \MakeUppercase{%
9591     #2{\glstextaccessdisplay
9592         {\glentrytext{\glslabel}}{\glslabel}}%
9593     {\glsdescriptionaccessdisplay
9594         {\glentrydesc{\glslabel}}{\glslabel}}%
9595     {\glssymbolaccessdisplay
9596         {\glentrysymbol{\glslabel}}{\glslabel}}%
9597     {\glsinsert}}%
9598 }%
9599 {%

```

First use

```

9600         \MakeUppercase{%
9601         #1{\glsfirstaccessdisplay
9602             {\glsentryfirst{\glslabel}}{\glslabel}}%
9603             {\glsdescriptionaccessdisplay
9604             {\glsentrydesc{\glslabel}}{\glslabel}}%
9605             {\glssymbolaccessdisplay
9606             {\glsentrysymbol{\glslabel}}{\glslabel}}%
9607             {\glsinsert}}%
9608     }%
9609 }%
9610 }%
9611 }%
9612 {%

```

Custom text provided in \glsdisp

```

9613     \ifglsused{\glslabel}%
9614     {%

```

Subsequent use

```

9615     #2{\glscustomtext}%
9616     {\glsdescriptionaccessdisplay
9617     {\glsentrydesc{\glslabel}}{\glslabel}}%
9618     {\glssymbolaccessdisplay
9619     {\glsentrysymbol{\glslabel}}{\glslabel}}%
9620     {\glsinsert}}%
9621 }%
9622 {%

```

First use

```

9623     #1{\glscustomtext}%
9624     {\glsdescriptionaccessdisplay
9625     {\glsentrydesc{\glslabel}}{\glslabel}}%
9626     {\glssymbolaccessdisplay
9627     {\glsentrysymbol{\glslabel}}{\glslabel}}%
9628     {\glsinsert}}%
9629 }%
9630 }%
9631 }

```

\glsгенentryfmt Redefine to use accessibility information.

```

9632 \renewcommand*{\glsгенentryfmt}{%
9633     \ifdefempty\glscustomtext
9634     {%
9635         \glsifplural
9636         {%

```

Plural form

```

9637         \glscapscase
9638         {%

```

Don't adjust case

```

9639      \ifglsused\glslabel
9640      {%

```

Subsequent use

```

9641      \glspluralaccessdisplay
9642      {\glsentryplural{\glslabel}}{\glslabel}%
9643      \glsinsert
9644      }%
9645      {%

```

First use

```

9646      \glsfirstpluralaccessdisplay
9647      {\glsentryfirstplural{\glslabel}}{\glslabel}%
9648      \glsinsert
9649      }%
9650      }%
9651      {%

```

Make first letter upper case

```

9652      \ifglsused\glslabel
9653      {%

```

Subsequent use.

```

9654      \glspluralaccessdisplay
9655      {\Glsentryplural{\glslabel}}{\glslabel}%
9656      \glsinsert
9657      }%
9658      {%

```

First use

```

9659      \glsfirstpluralaccessdisplay
9660      {\Glsentryfirstplural{\glslabel}}{\glslabel}%
9661      \glsinsert
9662      }%
9663      }%
9664      {%

```

Make all upper case

```

9665      \ifglsused\glslabel
9666      {%

```

Subsequent use

```

9667      \glspluralaccessdisplay
9668      {\mfirstucMakeUppercase{\glsentryplural{\glslabel}}}%
9669      {\glslabel}%
9670      \mfirstucMakeUppercase{\glsinsert}%
9671      }%
9672      {%

```

First use

```

9673      \glsfirstpluralaccessdisplay
9674      {\mfirstucMakeUppercase{\glsentryfirstplural{\glslabel}}}%

```

```

9675         {\glslabel}%
9676         \mfirstucMakeUppercase{\glsinsert}%
9677     }%
9678 }%
9679 }%
9680 {%

```

#### Singular form

```

9681     \glscapscase
9682     {%

```

#### Don't adjust case

```

9683     \ifglsused\glslabel
9684     {%

```

#### Subsequent use

```

9685     \glstextaccessdisplay{\glsentrytext{\glslabel}}{\glslabel}%
9686     \glsinsert
9687 }%
9688 {%

```

#### First use

```

9689     \glsfirstaccessdisplay{\glsentryfirst{\glslabel}}{\glslabel}%
9690     \glsinsert
9691 }%
9692 }%
9693 {%

```

#### Make first letter upper case

```

9694     \ifglsused\glslabel
9695     {%

```

#### Subsequent use

```

9696     \glstextaccessdisplay{\Glsentrytext{\glslabel}}{\glslabel}%
9697     \glsinsert
9698 }%
9699 {%

```

#### First use

```

9700     \glsfirstaccessdisplay{\Glsentryfirst{\glslabel}}{\glslabel}%
9701     \glsinsert
9702 }%
9703 }%
9704 {%

```

#### Make all upper case

```

9705     \ifglsused\glslabel
9706     {%

```

#### Subsequent use

```

9707     \glstextaccessdisplay
9708     {\mfirstucMakeUppercase{\glsentrytext{\glslabel}}}{\glslabel}%
9709     \mfirstucMakeUppercase{\glsinsert}%

```

```

9710         }%
9711         {%

First use
9712         \glsfirstaccessdisplay
9713         {\mfirstucMakeUppercase{\glstryfirst{\glslabel}}}{\glslabel}%
9714         \mfirstucMakeUppercase{\glsinsert}%
9715         }%
9716     }%
9717 }%
9718 }%
9719 {%

```

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.

```

9720     \glscustomtext\glsinsert
9721 }%
9722 }

```

`\glsacnfont` Redefine to include accessibility information.

```

9723 \renewcommand*{\glsacnfont}{%
9724     \ifempty\glscustomtext
9725     {%
9726         \ifglsused\glslabel
9727         {%

```

Subsequent use:

```

9728         \glsifplural
9729         {%

```

Subsequent plural form:

```

9730         \glscapscase
9731         {%

```

Subsequent plural form, don't adjust case:

```

9732         \acronymfont
9733         {\glsshortpluralaccessdisplay
9734         {\glstryshortpl{\glslabel}}{\glslabel}}%
9735         \glsinsert
9736     }%
9737     {%

```

Subsequent plural form, make first letter upper case:

```

9738         \acronymfont
9739         {\glsshortpluralaccessdisplay
9740         {\Glsentryshortpl{\glslabel}}{\glslabel}}%
9741         \glsinsert
9742     }%
9743     {%

```



Subsequent plural form, all caps:

```
9744      \mfirstucMakeUppercase
9745      {\acronymfont
9746      {\glsshortpluralaccessdisplay
9747      {\glentryshortpl{\glslabel}}{\glslabel}}%
9748      \glsinsert}%
9749      }%
9750      }%
9751      {%
```

Subsequent singular form

```
9752      \glscapscase
9753      {%
```

Subsequent singular form, don't adjust case:

```
9754      \acronymfont
9755      {\glsshortaccessdisplay{\glentryshort{\glslabel}}{\glslabel}}%
9756      \glsinsert
9757      }%
9758      {%
```

Subsequent singular form, make first letter upper case:

```
9759      \acronymfont
9760      {\glsshortaccessdisplay{\Glentryshort{\glslabel}}{\glslabel}}%
9761      \glsinsert
9762      }%
9763      {%
```

Subsequent singular form, all caps:

```
9764      \mfirstucMakeUppercase
9765      {\acronymfont{%
9766      \glsshortaccessdisplay{\glentryshort{\glslabel}}{\glslabel}}%
9767      \glsinsert}%
9768      }%
9769      }%
9770      }%
9771      {%
```

First use:

```
9772      \glsifplural
9773      {%
```

First use plural form:

```
9774      \glscapscase
9775      {%
```

First use plural form, don't adjust case:

```
9776      \genplacrfullformat{\glslabel}{\glsinsert}%
9777      }%
9778      {%
```

First use plural form, make first letter upper case:

```
9779      \Genplacrfullformat{\glslabel}{\glsinsert}%
9780      }%
9781      {%
```

First use plural form, all caps:

```
9782      \mfirstucMakeUppercase
9783      {\genplacrfullformat{\glslabel}{\glsinsert}}%
9784      }%
9785      }%
9786      {%
```

First use singular form

```
9787      \glscapscase
9788      {%
```

First use singular form, don't adjust case:

```
9789      \genacrfullformat{\glslabel}{\glsinsert}%
9790      }%
9791      {%
```

First use singular form, make first letter upper case:

```
9792      \Genacrfullformat{\glslabel}{\glsinsert}%
9793      }%
9794      {%
```

First use singular form, all caps:

```
9795      \mfirstucMakeUppercase
9796      {\genacrfullformat{\glslabel}{\glsinsert}}%
9797      }%
9798      }%
9799      }%
9800      }%
9801      {%
```

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.

```
9802      \glscustomtext
9803      }%
9804      }
```

`\genacrfullformat` Redefine to include accessibility information.

```
9805 \renewcommand*{\genacrfullformat}[2]{%
9806   \glslongaccessdisplay{\glsentrylong{#1}}{#1}#2\space
9807   (\glsshortaccessdisplay{\protect\firstacronymfont{\glsentryshort{#1}}}{#1})%
9808 }
```

`\Genacrfullformat` Redefine to include accessibility information.

```
9809 \renewcommand*{\Genacrfullformat}[2]{%
9810   \glslongaccessdisplay{\Glsentrylong{#1}}{#1}#2\space
9811   (\glsshortaccessdisplay{\protect\firstacronymfont{\Glsentryshort{#1}}}{#1})%
9812 }
```

\genplacrfullformat   Redefine to include accessibility information.

```
9813 \renewcommand*{\genplacrfullformat}[2]{%
9814   \glslongpluralaccessdisplay{\glsentrylongpl{#1}}{#1}#2\space
9815   (\glsshortpluralaccessdisplay
9816     {\protect\firstacronymfont{\glsentryshortpl{#1}}{#1}})%
9817 }
```

\Genplacrfullformat   Redefine to include accessibility information.

```
9818 \renewcommand*{\Genplacrfullformat}[2]{%
9819   \glslongpluralaccessdisplay{\Glsentrylongpl{#1}}{#1}#2\space
9820   (\glsshortpluralaccessdisplay
9821     {\protect\firstacronymfont{\glsentryshortpl{#1}}{#1}})%
9822 }
```

\@acrshort

```
9823 \def\@acrshort#1#2[#3]{%
9824   \glsdoifexists{#2}%
9825   {%
9826     \edef\@glo@type{\glsentrytype{#2}}%

9827     \let\glsifplural\@secondoftwo
9828     \let\glscapscase\@firstofthree
9829     \let\glsinsert\@empty
9830     \def\glscustomtext{%
9831       \acronymfont{\glsshortaccessdisplay{\glsentryshort{#2}}{#2}}#3%
9832     }%

```

Call \@gls@link

```
9833   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
9834   }%
9835 }
```

\@Acrshort

```
9836 \def\@Acrshort#1#2[#3]{%
9837   \glsdoifexists{#2}%
9838   {%
9839     \edef\@glo@type{\glsentrytype{#2}}%

9840     \let\glsifplural\@secondoftwo
9841     \let\glscapscase\@secondofthree
9842     \let\glsinsert\@empty
9843     \def\glscustomtext{%
9844       \acronymfont{\glsshortaccessdisplay{\Glsentryshort{#2}}{#2}}#3%
9845     }%

```

Call \@gls@link

```
9846   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
9847   }%
9848 }
```

\@ACRshort

```
9849 \def\@ACRshort#1#2[#3]{%
9850   \glsdoifexists{#2}%
9851   {%
9852     \edef\@glo@type{\glsentrytype{#2}}%

9853     \let\glsifplural\@secondoftwo
9854     \let\glscapscase\@thirdofthree
9855     \let\glsinsert\@empty
9856     \def\glscustomtext{%
9857       \acronymfont{\glsshortaccessdisplay
9858         {\MakeUppercase{\glsentryshort{#2}}}{#2}}#3%
9859     }%

    Call \@gls@link
9860     \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
9861   }%
9862 }
```

\@acrlong

```
9863 \def\@acrlong#1#2[#3]{%
9864   \glsdoifexists{#2}%
9865   {%
9866     \edef\@glo@type{\glsentrytype{#2}}%

9867     \let\glsifplural\@secondoftwo
9868     \let\glscapscase\@firstofthree
9869     \let\glsinsert\@empty
9870     \def\glscustomtext{%
9871       \acronymfont{\glslongaccessdisplay{\glsentrylong{#2}}{#2}}#3%
9872     }%

    Call \@gls@link
9873     \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
9874   }%
9875 }
```

\@Acrlong

```
9876 \def\@Acrlong#1#2[#3]{%
9877   \glsdoifexists{#2}%
9878   {%
9879     \edef\@glo@type{\glsentrytype{#2}}%

9880     \let\glsifplural\@secondoftwo
9881     \let\glscapscase\@firstofthree
9882     \let\glsinsert\@empty
9883     \def\glscustomtext{%
9884       \acronymfont{\glslongaccessdisplay{\Glsentrylong{#2}}{#2}}#3%
9885     }%

    Call \@gls@link
9886     \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
9887   }%
9888 }
```

```

Call \@gls@link
9886   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
9887   }%
9888 }

```

\@ACRlong

```

9889 \def\@ACRlong#1#2[#3]{%
9890   \glsdoifexists{#2}%
9891   {%
9892     \edef\@glo@type{\glstentrytype{#2}}%

9893     \let\glsifplural\@secondoftwo
9894     \let\glscapscase\@firstofthree
9895     \let\glsinsert\@empty
9896     \def\glscustomtext{%
9897       \acronymfont{\glslongaccessdisplay{%
9898         \MakeUppercase{\glstentrylong{#2}}}{#2}#3}%
9899     }%

```

```

Call \@gls@link
9900   \@gls@link[#1]{#2}{\csname gls@\@glo@type @entryfmt\endcsname}%
9901   }%
9902 }

```

### 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.

```

9903 \renewcommand*{\glossentryname}[1]{%
9904   \glsdoifexists{#1}%
9905   {%
9906     \glsnamefont{\glsnameaccessdisplay{\glstentryname{#1}}{#1}}%
9907   }%
9908 }

9909 \renewcommand*{\glossentryname}[1]{%
9910   \glsdoifexists{#1}%
9911   {%
9912     \glsnamefont{\glsnameaccessdisplay{\Glsentryname{#1}}{#1}}%
9913   }%
9914 }

9915 \renewcommand*{\glossentrydesc}[1]{%
9916   \glsdoifexists{#1}%

```

```

9917 {%
9918   \glsdescriptionaccessdisplay{\glsentrydesc{#1}}{#1}%
9919 }%
9920 }

9921 \renewcommand*{\Glossentrydesc}[1]{%
9922   \glsdoifexists{#1}%
9923   {%
9924     \glsdescriptionaccessdisplay{\Glsentrydesc{#1}}{#1}%
9925   }%
9926 }

9927 \renewcommand*{\glossentrysymbol}[1]{%
9928   \glsdoifexists{#1}%
9929   {%
9930     \glssymbolaccessdisplay{\glsentrysymbol{#1}}{#1}%
9931   }%
9932 }

9933 \renewcommand*{\Glossentrysymbol}[1]{%
9934   \glsdoifexists{#1}%
9935   {%
9936     \glssymbolaccessdisplay{\Glsentrysymbol{#1}}{#1}%
9937   }%
9938 }

```

pglossaryentryfield

```

9939 \newcommand*{\accsuppglossaryentryfield}[5]{%
9940   \glossaryentryfield{#1}%
9941   {\glsnameaccessdisplay{#2}{#1}}%
9942   {\glsdescriptionaccessdisplay{#3}{#1}}%
9943   {\glssymbolaccessdisplay{#4}{#1}}{#5}%
9944 }

```

glossarysubentryfield

```

9945 \newcommand*{\accsuppglossarysubentryfield}[6]{%
9946   \glossarysubentryfield{#1}{#2}%
9947   {\glsnameaccessdisplay{#3}{#2}}%
9948   {\glsdescriptionaccessdisplay{#4}{#2}}%
9949   {\glssymbolaccessdisplay{#5}{#2}}{#6}%
9950 }

```

## 6.4 Acronyms

Redefine acronym styles provided by glossaries:

long-short    *<long>* (*<short>*) acronym style.

```

9951 \renewacronymstyle{long-short}%
9952 {%

```

Check for long form in case this is a mixed glossary.

```

9953 \ifglshaslong{\glslabel}{\glsacronymfont}{\glsacronymentryfont}%
9954 }%
9955 {%
9956 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
9957 \renewcommand*{\genacrfullformat}[2]{%
9958 \glslongaccessdisplay{\glsentrylong{##1}}{##1}##2\space
9959 (\glsshortaccessdisplay
9960 {\protect\firstacronymfont{\glsentryshort{##1}}}{##1})%
9961 }%
9962 \renewcommand*{\Genacrfullformat}[2]{%
9963 \glslongaccessdisplay{\Glsentrylong{##1}}{##1}##2\space
9964 (\glsshortaccessdisplay
9965 {\protect\firstacronymfont{\glsentryshort{##1}}}{##1})%
9966 }%
9967 \renewcommand*{\genplacrfullformat}[2]{%
9968 \glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1}##2\space
9969 (\glsshortpluralaccessdisplay
9970 {\protect\firstacronymfont{\glsentryshortpl{##1}}}{##1})%
9971 }%
9972 \renewcommand*{\Genplacrfullformat}[2]{%
9973 \glslongpluralaccessdisplay{\Glsentrylongpl{##1}}{##1}##2\space
9974 (\glsshortpluralaccessdisplay
9975 {\protect\firstacronymfont{\glsentryshortpl{##1}}}{##1})%
9976 }%
9977 \renewcommand*{\acronymentry}[1]{%
9978 \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}}
9979 \renewcommand*{\acronymentry}[2]{##1}%
9980 \renewcommand*{\acronymfont}[1]{##1}%
9981 \renewcommand*{\firstacronymfont}[1]{\acronymfont{##1}}%
9982 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
9983 }

```

short-long <short> (<long>) acronym style.

```

9984 \renewacronymstyle{short-long}%
9985 {%

```

Check for long form in case this is a mixed glossary.

```

9986 \ifglshaslong{\glslabel}{\glsacronymfont}{\glsacronymentryfont}%
9987 }%
9988 {%
9989 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
9990 \renewcommand*{\genacrfullformat}[2]{%
9991 \glsshortaccessdisplay
9992 {\protect\firstacronymfont{\glsentryshort{##1}}}{##1}##2\space
9993 (\glslongaccessdisplay{\glsentrylong{##1}}{##1})%
9994 }%
9995 \renewcommand*{\Genacrfullformat}[2]{%
9996 \glsshortaccessdisplay

```

```

9997      {\protect\firstacronymfont{\Glsentryshort{##1}}}{##1}##2\space
9998      (\glslongaccessdisplay{\glsentrylong{##1}}{##1})%
9999  }%
10000 \renewcommand*{\genplacrfullformat}[2]{%
10001   \glsshortpluralaccessdisplay
10002   {\protect\firstacronymfont{\glsentryshortpl{##1}}}{##1}##2\space
10003   (\glslongpluralaccessdisplay
10004   {\glsentrylongpl{##1}}{##1})%
10005 }%
10006 \renewcommand*{\Genplacrfullformat}[2]{%
10007   \glsshortpluralaccessdisplay
10008   {\protect\firstacronymfont{\Glsentryshortpl{##1}}}{##1}##2\space
10009   (\glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1})%
10010 }%
10011 \renewcommand*{\acronymentry}[1]{%
10012   \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}%
10013 \renewcommand*{\acronymsort}[2]{##1}%
10014 \renewcommand*{\acronymfont}[1]{##1}%
10015 \renewcommand*{\firstacronymfont}[1]{\acronymfont{##1}}%
10016 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
10017 }

```

long-short-desc    *<long>* (*<short>*) acronym style that has an accompanying description (which the user needs to supply).

```

10018 \renewacronymstyle{long-short-desc}%
10019 {%
10020   \GlsUseAcrEntryDispStyle{long-short}%
10021 }%
10022 {%
10023   \GlsUseAcrStyleDefs{long-short}%
10024   \renewcommand*{\GenericAcronymFields}{}%
10025   \renewcommand*{\acronymsort}[2]{##2}%
10026   \renewcommand*{\acronymentry}[1]{%
10027     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10028     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10029 }

```

long-sc-short-desc    *<long>* (*\textsc{<short>}*) acronym style that has an accompanying description (which the user needs to supply).

```

10030 \renewacronymstyle{long-sc-short-desc}%
10031 {%
10032   \GlsUseAcrEntryDispStyle{long-sc-short}%
10033 }%
10034 {%
10035   \GlsUseAcrStyleDefs{long-sc-short}%
10036   \renewcommand*{\GenericAcronymFields}{}%
10037   \renewcommand*{\acronymsort}[2]{##2}%
10038   \renewcommand*{\acronymentry}[1]{%
10039     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space

```



```

10040      (\glsshortaccessdisplay{\acronymfont{\glentryshort{##1}}}{##1}})%
10041 }

```

long-sm-short-desc    *⟨long⟩* (\textsmaller{*⟨short⟩*}) acronym style that has an accompanying description (which the user needs to supply).

```

10042 \renewacronymstyle{long-sm-short-desc}%
10043 {%
10044   \GlsUseAcrEntryDispStyle{long-sm-short}%
10045 }%
10046 {%
10047   \GlsUseAcrStyleDefs{long-sm-short}%
10048   \renewcommand*{\GenericAcronymFields}{}%
10049   \renewcommand*{\acronymsort}[2]{##2}%
10050   \renewcommand*{\acronymentry}[1]{%
10051     \glslongaccessdisplay{\glentrylong{##1}}{##1}\space
10052     (\glsshortaccessdisplay{\acronymfont{\glentryshort{##1}}}{##1}})%
10053 }

```

short-long-desc    *⟨short⟩* ({*⟨long⟩*}) acronym style that has an accompanying description (which the user needs to supply).

```

10054 \renewacronymstyle{short-long-desc}%
10055 {%
10056   \GlsUseAcrEntryDispStyle{short-long}%
10057 }%
10058 {%
10059   \GlsUseAcrStyleDefs{short-long}%
10060   \renewcommand*{\GenericAcronymFields}{}%
10061   \renewcommand*{\acronymsort}[2]{##2}%
10062   \renewcommand*{\acronymentry}[1]{%
10063     \glslongaccessdisplay{\glentrylong{##1}}{##1}\space
10064     (\glsshortaccessdisplay{\acronymfont{\glentryshort{##1}}}{##1}})%
10065 }

```

sc-short-long-desc    *⟨long⟩* (\textsc{*⟨short⟩*}) acronym style that has an accompanying description (which the user needs to supply).

```

10066 \renewacronymstyle{sc-short-long-desc}%
10067 {%
10068   \GlsUseAcrEntryDispStyle{sc-short-long}%
10069 }%
10070 {%
10071   \GlsUseAcrStyleDefs{sc-short-long}%
10072   \renewcommand*{\GenericAcronymFields}{}%
10073   \renewcommand*{\acronymsort}[2]{##2}%
10074   \renewcommand*{\acronymentry}[1]{%
10075     \glslongaccessdisplay{\glentrylong{##1}}{##1}\space
10076     (\glsshortaccessdisplay{\acronymfont{\glentryshort{##1}}}{##1}})%
10077 }

```

```
sm-short-long-desc  <long> (\textsmaller{<short>}) acronym style that has an accompanying de-
                    scription (which the user needs to supply).
10078 \renewacronymstyle{sm-short-long-desc}%
10079 {%
10080   \GlsUseAcrEntryDispStyle{sm-short-long}%
10081 }%
10082 {%
10083   \GlsUseAcrStyleDefs{sm-short-long}%
10084   \renewcommand*{\GenericAcronymFields}{}%
10085   \renewcommand*{\acronymsort}[2]{##2}%
10086   \renewcommand*{\acronymentry}[1]{%
10087     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10088     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10089 }
```

dua <long> only acronym style.

```
10090 \renewacronymstyle{dua}%
10091 {%
```

Check for long form in case this is a mixed glossary.

```
10092 \ifdefempty\glscustomtext
10093 {%
10094   \ifglshaslong{\glslabel}%
10095   {%
10096     \glsifplural
10097     {%
```

Plural form:

```
10098     \glscapscase
10099     {%
```

Plural form, don't adjust case:

```
10100     \glslongpluralaccessdisplay{\glsentrylongpl{\glslabel}}{\glslabel}%
10101     \glsinsert
10102   }%
10103   {%
```

Plural form, make first letter upper case:

```
10104     \glslongpluralaccessdisplay{\Glsentrylongpl{\glslabel}}{\glslabel}%
10105     \glsinsert
10106   }%
10107   {%
```

Plural form, all caps:

```
10108     \glslongpluralaccessdisplay
10109     {\mfirstucMakeUppercase{\glsentrylongpl{\glslabel}}}{\glslabel}%
10110     \mfirstucMakeUppercase{\glsinsert}%
10111   }%
10112   }%
10113   {%
```

### Singular form

```
10114      \glscapscase
10115      {%
```

### Singular form, don't adjust case:

```
10116      \glslongaccessdisplay{\glsentrylong{\glslabel}}{\glslabel}\glsinsert
10117      }%
10118      {%
```

### Subsequent singular form, make first letter upper case:

```
10119      \glslongaccessdisplay{\Glsentrylong{\glslabel}}{\glslabel}\glsinsert
10120      }%
10121      {%
```

### Subsequent singular form, all caps:

```
10122      \glslongaccessdisplay
10123      {\mfirstucMakeUppercase
10124      {\glsentrylong{\glslabel}\glsinsert}}{\glslabel}%
10125      \mfirstucMakeUppercase{\glsinsert}%
10126      }%
10127      }%
10128      }%
10129      {%
```

### Not an acronym:

```
10130      \glsgenentryfmt
10131      }%
10132      }%
10133      {\glscustomtext\glsinsert}%
10134      }%
10135      {%
10136      \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
10137      \renewcommand*{\acrfullfmt}[3]{%
10138      \glslink[##1]{##2}{%
10139      \glslongaccessdisplay{\glsentrylong{##2}}{##2}##3\space
10140      (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##2}}}{##2})}%
10141      \renewcommand*{\Acrfullfmt}[3]{%
10142      \glslink[##1]{##2}{%
10143      \glslongaccessdisplay{\Glsentrylong{##2}}{##2}##3\space
10144      (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##2}}}{##2})}%
10145      \renewcommand*{\ACRfullfmt}[3]{%
10146      \glslink[##1]{##2}{%
10147      \glslongaccessdisplay
10148      {\mfirstucMakeUppercase{\glsentrylong{##2}}{##2}##3\space
10149      (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##2}}}{##2})}}}%
10150      \renewcommand*{\acrfullplfmt}[3]{%
10151      \glslink[##1]{##2}{%
10152      \glslongpluralaccessdisplay
10153      {\glsentrylongpl{##2}}{##2}##3\space
10154      (\glsshortpluralaccessdisplay
10155      {\acronymfont{\glsentryshortpl{##2}}}{##2})}%
```

```

10156 \renewcommand*{\Acrfullplfmt}[3]{%
10157   \glslink[##1]{##2}{%
10158     \glslongpluralaccessdisplay
10159     {\Glsentrylongpl{##2}}{##2}##3\space
10160     (\glsshortpluralaccessdisplay
10161     {\acronymfont{\glsentryshortpl{##2}}{##2}})}%
10162 \renewcommand*{\ACRfullplfmt}[3]{%
10163   \glslink[##1]{##2}{%
10164     \glslongpluralaccessdisplay
10165     {\mfirstucMakeUppercase{\glsentrylongpl{##2}}{##2}##3\space
10166     (\glsshortpluralaccessdisplay
10167     {\acronymfont{\glsentryshortpl{##2}}{##2}})}%
10168 \renewcommand*{\glsentryfull}[1]{%
10169   \glslongaccessdisplay{\glsentrylong{##1}}\space
10170   (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}{##1}})%
10171 }%
10172 \renewcommand*{\Glsentryfull}[1]{%
10173   \glslongaccessdisplay{\Glsentrylong{##1}}{##1}\space
10174   (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}{##1}})%
10175 }%
10176 \renewcommand*{\glsentryfullpl}[1]{%
10177   \glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1}\space
10178   (\glsshortpluralaccessdisplay{\acronymfont{\glsentryshortpl{##1}}{##1}})%
10179 }%
10180 \renewcommand*{\Glsentryfullpl}[1]{%
10181   \glslongpluralaccessdisplay{\Glsentrylongpl{##1}}{##1}\space
10182   (\glsshortpluralaccessdisplay{\acronymfont{\glsentryshortpl{##1}}{##1}})%
10183 }%
10184 \renewcommand*{\acronymentry}[1]{%
10185   \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}{##1}}%
10186 \renewcommand*{\acronymsort}[2]{##1}%
10187 \renewcommand*{\acronymfont}[1]{##1}%
10188 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
10189 }

```

dua-desc *<long>* only acronym style with user-supplied description.

```

10190 \renewacronymstyle{dua-desc}%
10191 {%
10192   \GlsUseAcrEntryDispStyle{dua}%
10193 }%
10194 {%
10195   \GlsUseAcrStyleDefs{dua}%
10196 \renewcommand*{\GenericAcronymFields}{}%
10197 \renewcommand*{\acronymentry}[1]{%
10198   \glslongaccessdisplay{\acronymfont{\glsentrylong{##1}}{##1}}%
10199 \renewcommand*{\acronymsort}[2]{##2}%
10200 }%

```

footnote *<short>*\footnote{*<long>*} acronym style.

10201 \renewacronymstyle{footnote}%

10202 {%

Check for long form in case this is a mixed glossary.

10203 \ifglshaslong{\glslabel}{\glsngenacfmt}{\glsngenentryfmt}%

10204 }%

10205 {%

10206 \renewcommand\*{\GenericAcronymFields}{description={\the\glslongtok}}%

Need to ensure hyperlinks are switched off on first use:

10207 \glshyperfirstfalse

10208 \renewcommand\*{\genacrfullformat}[2]{%

10209 \glsshortaccessdisplay

10210 {\protect\firstacronymfont{\glsentryshort{##1}}}{##1}##2%

10211 \protect\footnote{\glslongaccessdisplay{\glsentrylong{##1}}}{##1}}%

10212 }%

10213 \renewcommand\*{\Genacrfullformat}[2]{%

10214 \glsshortaccessdisplay

10215 {\firstacronymfont{\Glsentryshort{##1}}}{##1}##2%

10216 \protect\footnote{\glslongaccessdisplay{\glsentrylong{##1}}}{##1}}%

10217 }%

10218 \renewcommand\*{\genplacrfullformat}[2]{%

10219 \glsshortpluralaccessdisplay

10220 {\protect\firstacronymfont{\glsentryshortpl{##1}}}{##1}##2%

10221 \protect\footnote{\glslongpluralaccessdisplay{\glsentrylongpl{##1}}}{##1}}%

10222 }%

10223 \renewcommand\*{\Genplacrfullformat}[2]{%

10224 \glsshortpluralaccessdisplay

10225 {\protect\firstacronymfont{\Glsentryshortpl{##1}}}{##1}##2%

10226 \protect\footnote{\glslongpluralaccessdisplay{\glsentrylongpl{##1}}}{##1}}%

10227 }%

10228 \renewcommand\*{\acronymentry}[1]{%

10229 \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}}%

10230 \renewcommand\*{\acronymsort}[2]{##1}%

10231 \renewcommand\*{\acronymfont}[1]{##1}%

10232 \renewcommand\*{\acrpluralsuffix}{\glspluralsuffix}%

Don't use footnotes for \acrfull:

10233 \renewcommand\*{\acrfullfmt}[3]{%

10234 \glslink[##1]{##2}{%

10235 \glsshortaccessdisplay{\acronymfont{\glsentryshort{##2}}}{##2}##3\space

10236 (\glslongaccessdisplay{\glsentrylong{##2}}{##2}})%

10237 \renewcommand\*{\Acrfullfmt}[3]{%

10238 \glslink[##1]{##2}{%

10239 \glsshortaccessdisplay{\acronymfont{\Glsentryshort{##2}}}{##2}##3\space

10240 (\glslongaccessdisplay{\glsentrylong{##2}}{##2}})%

10241 \renewcommand\*{\ACRfullfmt}[3]{%

10242 \glslink[##1]{##2}{%

10243 \glsshortaccessdisplay

10244 {\mfirstucMakeUppercase

10245 {\acronymfont{\glsentryshort{##2}}}{##2}##3\space

```

10246      (\glslongaccessdisplay{\glsentrylong{##2}}{##2}})}%
10247 \renewcommand*{\acrfullplfmt}[3]{%
10248   \glslink[##1]{##2}{%
10249     \glsshortpluralaccessdisplay
10250     {\acronymfont{\glsentryshortpl{##2}}{##2}##3\space
10251     (\glslongpluralaccessdisplay{\glsentrylongpl{##2}}{##2}})}%
10252 \renewcommand*{\Acrfullplfmt}[3]{%
10253   \glslink[##1]{##2}{%
10254     \glsshortpluralaccessdisplay
10255     {\acronymfont{\Glsentryshortpl{##2}}{##2}##3\space
10256     (\glslongpluralaccessdisplay{\glsentrylongpl{##2}})}%
10257 \renewcommand*{\ACRfullplfmt}[3]{%
10258   \glslink[##1]{##2}{%
10259     \glsshortpluralaccessdisplay
10260     {\mfirstucMakeUppercase
10261     {\acronymfont{\glsentryshortpl{##2}}{##2}##3\space
10262     (\glslongpluralaccessdisplay{\glsentrylongpl{##2}}{##2}})}%

```

Similarly for \glsentryfull etc:

```

10263 \renewcommand*{\glsentryfull}[1]{%
10264   \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}{##1}\space
10265   (\glslongaccessdisplay{\glsentrylong{##1}}{##1}})%
10266 \renewcommand*{\Glsentryfull}[1]{%
10267   \glsshortaccessdisplay{\acronymfont{\Glsentryshort{##1}}{##1}\space
10268   (\glslongaccessdisplay{\glsentrylong{##1}}{##1}})%
10269 \renewcommand*{\glsentryfullpl}[1]{%
10270   \glsshortpluralaccessdisplay
10271   {\acronymfont{\glsentryshortpl{##1}}{##1}\space
10272   (\glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1}})%
10273 \renewcommand*{\Glsentryfullpl}[1]{%
10274   \glsshortpluralaccessdisplay
10275   {\acronymfont{\Glsentryshortpl{##1}}{##1}\space
10276   (\glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1}})%
10277 }

```

footnote-sc \textsc{<short>}\footnote{<long>} acronym style.

```

10278 \renewacronymstyle{footnote-sc}%
10279 {%
10280   \GlsUseAcrEntryDisplayStyle{footnote}%
10281 }%
10282 {%
10283   \GlsUseAcrStyleDefs{footnote}%
10284   \renewcommand{\acronymentry}[1]{%
10285     \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}{##1}}
10286     \renewcommand{\acronymfont}[1]{\textsc{##1}}%
10287     \renewcommand*{\acrpluralsuffix}{\glstextup{\glspluralsuffix}}%
10288 }%

```

footnote-sm \textsmaller{<short>}\footnote{<long>} acronym style.

```

10289 \renewacronymstyle{footnote-sm}%
10290 {%
10291   \GlsUseAcrEntryDispStyle{footnote}%
10292 }%
10293 {%
10294   \GlsUseAcrStyleDefs{footnote}%
10295   \renewcommand{\acronymentry}[1]{%
10296     \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}}
10297   \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
10298   \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
10299 }%

```

footnote-desc *<short>*\footnote{*<long>*} acronym style that has an accompanying description (which the user needs to supply).

```

10300 \renewacronymstyle{footnote-desc}%
10301 {%
10302   \GlsUseAcrEntryDispStyle{footnote}%
10303 }%
10304 {%
10305   \GlsUseAcrStyleDefs{footnote}%
10306   \renewcommand*{\GenericAcronymFields}{}%
10307   \renewcommand*{\acronymsort}[2]{##2}%
10308   \renewcommand*{\acronymentry}[1]{%
10309     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10310     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1))}%
10311 }

```

footnote-sc-desc \textsc{*<short>*}\footnote{*<long>*} acronym style that has an accompanying description (which the user needs to supply).

```

10312 \renewacronymstyle{footnote-sc-desc}%
10313 {%
10314   \GlsUseAcrEntryDispStyle{footnote-sc}%
10315 }%
10316 {%
10317   \GlsUseAcrStyleDefs{footnote-sc}%
10318   \renewcommand*{\GenericAcronymFields}{}%
10319   \renewcommand*{\acronymsort}[2]{##2}%
10320   \renewcommand*{\acronymentry}[1]{%
10321     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10322     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1))}%
10323 }

```

footnote-sm-desc \textsmaller{*<short>*}\footnote{*<long>*} acronym style that has an accompanying description (which the user needs to supply).

```

10324 \renewacronymstyle{footnote-sm-desc}%
10325 {%
10326   \GlsUseAcrEntryDispStyle{footnote-sm}%
10327 }%
10328 {%

```

```

10329 \GlsUseAcrStyleDefs{footnote-sm}%
10330 \renewcommand*{\GenericAcronymFields}{}%
10331 \renewcommand*{\acronymsort}[2]{##2}%
10332 \renewcommand*{\acronymentry}[1]{%
10333   \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10334   (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10335 }

```

Use `\newacronymhook` to modify the key list to set the access text to the long version by default.

```

10336 \renewcommand*{\newacronymhook}{%
10337   \edef\@gls@keylist{shortaccess=\the\glslongtok,%
10338     \the\glskeylisttok}%
10339   \expandafter\glskeylisttok\expandafter{\@gls@keylist}%
10340 }

```

`\defaultNewAcronymDef` Modify default style to use access text:

```

10341 \renewcommand*{\DefaultNewAcronymDef}{%
10342   \edef\@do@newglossaryentry{%
10343     \noexpand\newglossaryentry{\the\glslabeltok}%
10344     {%
10345       type=\acronymtype,%
10346       name={\the\glsshorttok},%
10347       description={\the\glslongtok},%
10348       descriptionaccess=\relax,
10349       text={\the\glsshorttok},%
10350       access={\noexpand\@glo@textaccess},%
10351       sort={\the\glsshorttok},%
10352       short={\the\glsshorttok},%
10353       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10354       shortaccess={\the\glslongtok},%
10355       long={\the\glslongtok},%
10356       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10357       descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10358       first={\noexpand\glslongaccessdisplay
10359         {\the\glslongtok}{\the\glslabeltok}\space
10360         (\noexpand\glsshortaccessdisplay
10361           {\the\glsshorttok}{\the\glslabeltok})},%
10362       plural={\the\glsshorttok\acrpluralsuffix},%
10363       firstplural={\noexpand\glslongpluralaccessdisplay
10364         {\noexpand\@glo@longpl}{\the\glslabeltok}\space
10365         (\noexpand\glsshortpluralaccessdisplay
10366           {\noexpand\@glo@shortpl}{\the\glslabeltok})},%
10367       firstaccess=\relax,
10368       firstpluralaccess=\relax,
10369       textaccess={\noexpand\@glo@shortaccess},%
10370       \the\glskeylisttok
10371     }%
10372   }%

```



```

10373 \let\@org@gl@s@assign@firstpl\gl@s@assign@firstpl
10374 \let\@org@gl@s@assign@plural\gl@s@assign@plural
10375 \let\@org@gl@s@assign@descplural\gl@s@assign@descplural
10376 \def\gl@s@assign@firstpl##1##2{%
10377   \@@gl@s@expand@field{##1}{firstpl}{##2}%
10378 }%
10379 \def\gl@s@assign@plural##1##2{%
10380   \@@gl@s@expand@field{##1}{plural}{##2}%
10381 }%
10382 \def\gl@s@assign@descplural##1##2{%
10383   \@@gl@s@expand@field{##1}{descplural}{##2}%
10384 }%
10385 \do@newglossaryentry
10386 \let\gl@s@assign@firstpl\@org@gl@s@assign@firstpl
10387 \let\gl@s@assign@plural\@org@gl@s@assign@plural
10388 \let\gl@s@assign@symbolplural\@org@gl@s@assign@symbolplural
10389 }

```

otnoteNewAcronymDef

```

10390 \renewcommand*{\DescriptionFootnoteNewAcronymDef}{%
10391   \edef\@do@newglossaryentry{%
10392     \noexpand\newglossaryentry{\the\glslabelltok}%
10393     {%
10394       type=\acronymtype,%
10395       name={\noexpand\acronymfont{\the\glsshorttok}},%
10396       sort={\the\glsshorttok},%
10397       text={\the\glsshorttok},%
10398       short={\the\glsshorttok},%
10399       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10400       shortaccess={\the\glslongtok},%
10401       long={\the\glslongtok},%
10402       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10403       access={\noexpand\@glo@textaccess},%
10404       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10405       symbol={\the\glslongtok},%
10406       symbolplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10407       firstpluralaccess=\relax,
10408       textaccess={\noexpand\@glo@shortaccess},%
10409       \the\glscopylisttok
10410     }%
10411   }%
10412   \let\@org@gl@s@assign@firstpl\gl@s@assign@firstpl
10413   \let\@org@gl@s@assign@plural\gl@s@assign@plural
10414   \let\@org@gl@s@assign@symbolplural\gl@s@assign@symbolplural
10415   \def\gl@s@assign@firstpl##1##2{%
10416     \@@gl@s@expand@field{##1}{firstpl}{##2}%
10417   }%
10418   \def\gl@s@assign@plural##1##2{%
10419     \@@gl@s@expand@field{##1}{plural}{##2}%

```

```

10420 }%
10421 \def\gls@assign@symbolplural##1##2{%
10422   \@@gls@expand@field{##1}{symbolplural}{##2}%
10423 }%
10424 \@do@newglossaryentry
10425 \let\gls@assign@plural\@org@gls@assign@plural
10426 \let\gls@assign@firstpl\@org@gls@assign@firstpl
10427 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
10428 }

```

ptionNewAcronymDef

```

10429 \renewcommand*{\DescriptionNewAcronymDef}{%
10430   \edef\@do@newglossaryentry{%
10431     \noexpand\newglossaryentry{\the\glslabeltok}%
10432     {%
10433       type=\acronymtype,%
10434       name={\noexpand
10435         \acrnameformat{\the\glsshorttok}{\the\glslongtok}},%
10436       access={\noexpand\@glo@textaccess},%
10437       sort={\the\glsshorttok},%
10438       short={\the\glsshorttok},%
10439       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10440       shortaccess={\the\glslongtok},%
10441       long={\the\glslongtok},%
10442       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10443       first={\the\glslongtok},%
10444       firstaccess=\relax,
10445       firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10446       text={\the\glsshorttok},%
10447       textaccess={\the\glslongtok},%
10448       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10449       symbol={\noexpand\@glo@text},%
10450       symbolaccess={\noexpand\@glo@textaccess},%
10451       symbolplural={\noexpand\@glo@plural},%
10452       firstpluralaccess=\relax,
10453       textaccess={\noexpand\@glo@shortaccess},%
10454       \the\glskeylisttok}%
10455   }%
10456   \let\@org@gls@assign@firstpl\gls@assign@firstpl
10457   \let\@org@gls@assign@plural\gls@assign@plural
10458   \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
10459   \def\gls@assign@firstpl##1##2{%
10460     \@@gls@expand@field{##1}{firstpl}{##2}%
10461   }%
10462   \def\gls@assign@plural##1##2{%
10463     \@@gls@expand@field{##1}{plural}{##2}%
10464   }%
10465   \def\gls@assign@symbolplural##1##2{%
10466     \@@gls@expand@field{##1}{symbolplural}{##2}%

```

```

10467 }%
10468 \@do@newglossaryentry
10469 \let\gls@assign@firstpl\@org@gls@assign@firstpl
10470 \let\gls@assign@plural\@org@gls@assign@plural
10471 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
10472 }

```

otnoteNewAcronymDef

```

10473 \renewcommand*{\FootnoteNewAcronymDef}{%
10474   \edef\@do@newglossaryentry{%
10475     \noexpand\newglossaryentry{\the\glslabeltok}%
10476     {%
10477       type=\acronymtype,%
10478       name={\noexpand\acronymfont{\the\glsshorttok}},%
10479       sort={\the\glsshorttok},%
10480       text={\the\glsshorttok},%
10481       textaccess={\the\glslongtok},%
10482       access={\noexpand\@glo@textaccess},%
10483       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10484       short={\the\glsshorttok},%
10485       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10486       long={\the\glslongtok},%
10487       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10488       description={\the\glslongtok},%
10489       descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10490       \the\glskeylisttok
10491     }%
10492   }%
10493   \let\@org@gls@assign@plural\gls@assign@plural
10494   \let\@org@gls@assign@firstpl\gls@assign@firstpl
10495   \let\@org@gls@assign@descplural\gls@assign@descplural
10496   \def\gls@assign@firstpl##1##2{%
10497     \@@gls@expand@field{##1}{firstpl}{##2}%
10498   }%
10499   \def\gls@assign@plural##1##2{%
10500     \@@gls@expand@field{##1}{plural}{##2}%
10501   }%
10502   \def\gls@assign@descplural##1##2{%
10503     \@@gls@expand@field{##1}{descplural}{##2}%
10504   }%
10505   \@do@newglossaryentry
10506   \let\gls@assign@plural\@org@gls@assign@plural
10507   \let\gls@assign@firstpl\@org@gls@assign@firstpl
10508   \let\gls@assign@descplural\@org@gls@assign@descplural
10509 }

```

\SmallNewAcronymDef

```

10510 \renewcommand*{\SmallNewAcronymDef}{%
10511   \edef\@do@newglossaryentry{%

```

```

10512 \noexpand\newglossaryentry{\the\glslabeltok}%
10513 {%
10514     type=\acronymtype,%
10515     name={\noexpand\acronymfont{\the\glsshorttok}},%
10516     access={\noexpand\@glo@symbolaccess},%
10517     sort={\the\glsshorttok},%
10518     short={\the\glsshorttok},%
10519     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10520     shortaccess={\the\glslongtok},%
10521     long={\the\glslongtok},%
10522     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10523     text={\noexpand\@glo@short},%
10524     textaccess={\noexpand\@glo@shortaccess},%
10525     plural={\noexpand\@glo@shortpl},%
10526     first={\the\glslongtok},%
10527     firstaccess=\relax,%
10528     firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10529     description={\noexpand\@glo@first},%
10530     descriptionplural={\noexpand\@glo@firstplural},%
10531     symbol={\the\glsshorttok},%
10532     symbolaccess={\the\glslongtok},%
10533     symbolplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10534     \the\glskeylisttok
10535 }%
10536 }%
10537 \let\@org@gls@assign@firstpl\gls@assign@firstpl
10538 \let\@org@gls@assign@plural\gls@assign@plural
10539 \let\@org@gls@assign@descplural\gls@assign@descplural
10540 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
10541 \def\gls@assign@firstpl##1##2{%
10542     \@@gls@expand@field{##1}{firstpl}{##2}%
10543 }%
10544 \def\gls@assign@plural##1##2{%
10545     \@@gls@expand@field{##1}{plural}{##2}%
10546 }%
10547 \def\gls@assign@descplural##1##2{%
10548     \@@gls@expand@field{##1}{descplural}{##2}%
10549 }%
10550 \def\gls@assign@symbolplural##1##2{%
10551     \@@gls@expand@field{##1}{symbolplural}{##2}%
10552 }%
10553 \@do@newglossaryentry
10554 \let\gls@assign@firstpl\@org@gls@assign@firstpl
10555 \let\gls@assign@plural\@org@gls@assign@plural
10556 \let\gls@assign@descplural\@org@gls@assign@descplural
10557 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
10558 }

```

The following are kept for compatibility with versions before 3.0:

```

\glsshortaccesskey
10559 \newcommand*{\glsshortaccesskey}{\glsshortkey access}%

hortpluralaccesskey
10560 \newcommand*{\glsshortpluralaccesskey}{\glsshortpluralkey access}%

\glslongaccesskey
10561 \newcommand*{\glslongaccesskey}{\glslongkey access}%

longpluralaccesskey
10562 \newcommand*{\glslongpluralaccesskey}{\glslongpluralkey access}%

```

## 6.5 Debugging Commands

```

\showglonameaccess
10563 \newcommand*{\showglonameaccess}[1]{%
10564 \expandafter\show\csname glo@\glsdetoklabel{#1}@textaccess\endcsname
10565 }

\showglotextaccess
10566 \newcommand*{\showglotextaccess}[1]{%
10567 \expandafter\show\csname glo@\glsdetoklabel{#1}@textaccess\endcsname
10568 }

showglopluralaccess
10569 \newcommand*{\showglopluralaccess}[1]{%
10570 \expandafter\show\csname glo@\glsdetoklabel{#1}@pluralaccess\endcsname
10571 }

\showglofirstaccess
10572 \newcommand*{\showglofirstaccess}[1]{%
10573 \expandafter\show\csname glo@\glsdetoklabel{#1}@firstaccess\endcsname
10574 }

lofirstpluralaccess
10575 \newcommand*{\showglofirstpluralaccess}[1]{%
10576 \expandafter\show\csname glo@\glsdetoklabel{#1}@firstpluralaccess\endcsname
10577 }

showglosymbolaccess
10578 \newcommand*{\showglosymbolaccess}[1]{%
10579 \expandafter\show\csname glo@\glsdetoklabel{#1}@symbolaccess\endcsname
10580 }

osymbolpluralaccess
10581 \newcommand*{\showglosymbolpluralaccess}[1]{%
10582 \expandafter\show\csname glo@\glsdetoklabel{#1}@symbolpluralaccess\endcsname
10583 }

```

```

\showglodescaccess
10584 \newcommand*\showglodescaccess}[1]{%
10585   \expandafter\show\csname glo@\glsdetoklabel{#1}@descaccess\endcsname
10586 }

glodescpluralaccess
10587 \newcommand*\showglodescpluralaccess}[1]{%
10588   \expandafter\show\csname glo@\glsdetoklabel{#1}@descpluralaccess\endcsname
10589 }

\showgloshortaccess
10590 \newcommand*\showgloshortaccess}[1]{%
10591   \expandafter\show\csname glo@\glsdetoklabel{#1}@shortaccess\endcsname
10592 }

loshortpluralaccess
10593 \newcommand*\showgloshortpluralaccess}[1]{%
10594   \expandafter\show\csname glo@\glsdetoklabel{#1}@shortpluralaccess\endcsname
10595 }

\showglolongaccess
10596 \newcommand*\showglolongaccess}[1]{%
10597   \expandafter\show\csname glo@\glsdetoklabel{#1}@longaccess\endcsname
10598 }

glolongpluralaccess
10599 \newcommand*\showglolongpluralaccess}[1]{%
10600   \expandafter\show\csname glo@\glsdetoklabel{#1}@longpluralaccess\endcsname
10601 }

```

## 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.

```

10602 \NeedsTeXFormat{LaTeX2e}
10603 \ProvidesPackage{glossaries-babel}[2013/11/14 v4.0 (NLCT)]

English:
10604 \@ifundefined{captionsenglish}{}{%
10605   \addto\captionsenglish{%
10606     \renewcommand*\glossaryname{Glossary}%
10607     \renewcommand*\acronymname{Acronyms}%

```

```

10608 \renewcommand*{\entryname}{Notation}%
10609 \renewcommand*{\descriptionname}{Description}%
10610 \renewcommand*{\symbolname}{Symbol}%
10611 \renewcommand*{\pagelistname}{Page List}%
10612 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10613 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10614 }%
10615 }
10616 \@ifundefined{captionsamerican}{}{%
10617 \addto\captionsamerican{%
10618 \renewcommand*{\glossaryname}{Glossary}%
10619 \renewcommand*{\acronymname}{Acronyms}%
10620 \renewcommand*{\entryname}{Notation}%
10621 \renewcommand*{\descriptionname}{Description}%
10622 \renewcommand*{\symbolname}{Symbol}%
10623 \renewcommand*{\pagelistname}{Page List}%
10624 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10625 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10626 }%
10627 }
10628 \@ifundefined{captionsaustralian}{}{%
10629 \addto\captionsaustralian{%
10630 \renewcommand*{\glossaryname}{Glossary}%
10631 \renewcommand*{\acronymname}{Acronyms}%
10632 \renewcommand*{\entryname}{Notation}%
10633 \renewcommand*{\descriptionname}{Description}%
10634 \renewcommand*{\symbolname}{Symbol}%
10635 \renewcommand*{\pagelistname}{Page List}%
10636 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10637 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10638 }%
10639 }
10640 \@ifundefined{captionsbritish}{}{%
10641 \addto\captionsbritish{%
10642 \renewcommand*{\glossaryname}{Glossary}%
10643 \renewcommand*{\acronymname}{Acronyms}%
10644 \renewcommand*{\entryname}{Notation}%
10645 \renewcommand*{\descriptionname}{Description}%
10646 \renewcommand*{\symbolname}{Symbol}%
10647 \renewcommand*{\pagelistname}{Page List}%
10648 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10649 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10650 }%
10651 \@ifundefined{captionscanadian}{}{%
10652 \addto\captionscanadian{%
10653 \renewcommand*{\glossaryname}{Glossary}%
10654 \renewcommand*{\acronymname}{Acronyms}%
10655 \renewcommand*{\entryname}{Notation}%
10656 \renewcommand*{\descriptionname}{Description}%

```

```

10657 \renewcommand*{\symbolname}{Symbol}%
10658 \renewcommand*{\pagelistname}{Page List}%
10659 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10660 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10661 }%
10662 }
10663 \@ifundefined{captionsnewzealand}{}{%
10664 \addto\captionsnewzealand{%
10665 \renewcommand*{\glossaryname}{Glossary}%
10666 \renewcommand*{\acronymname}{Acronyms}%
10667 \renewcommand*{\entryname}{Notation}%
10668 \renewcommand*{\descriptionname}{Description}%
10669 \renewcommand*{\symbolname}{Symbol}%
10670 \renewcommand*{\pagelistname}{Page List}%
10671 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10672 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10673 }%
10674 }
10675 \@ifundefined{captionsUKenglish}{}{%
10676 \addto\captionsUKenglish{%
10677 \renewcommand*{\glossaryname}{Glossary}%
10678 \renewcommand*{\acronymname}{Acronyms}%
10679 \renewcommand*{\entryname}{Notation}%
10680 \renewcommand*{\descriptionname}{Description}%
10681 \renewcommand*{\symbolname}{Symbol}%
10682 \renewcommand*{\pagelistname}{Page List}%
10683 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10684 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10685 }%
10686 }
10687 \@ifundefined{captionsUSenglish}{}{%
10688 \addto\captionsUSenglish{%
10689 \renewcommand*{\glossaryname}{Glossary}%
10690 \renewcommand*{\acronymname}{Acronyms}%
10691 \renewcommand*{\entryname}{Notation}%
10692 \renewcommand*{\descriptionname}{Description}%
10693 \renewcommand*{\symbolname}{Symbol}%
10694 \renewcommand*{\pagelistname}{Page List}%
10695 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10696 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10697 }%
10698 }

```

German (quite a few variations were suggested for German; I settled on the following):

```

10699 \@ifundefined{captionsgerman}{}{%
10700 \addto\captionsgerman{%
10701 \renewcommand*{\glossaryname}{Glossar}%
10702 \renewcommand*{\acronymname}{Akronyme}%
10703 \renewcommand*{\entryname}{Bezeichnung}%

```



```

10704 \renewcommand*{\descriptionname}{Beschreibung}%
10705 \renewcommand*{\symbolname}{Symbol}%
10706 \renewcommand*{\pagelistname}{Seiten}%
10707 \renewcommand*{\glssymbolsgroupname}{Symbole}%
10708 \renewcommand*{\glsnumbersgroupname}{Zahlen}}
10709 }

```

ngerman is identical to German:

```

10710 \@ifundefined{captionsngerman}{}{%
10711 \addto\captionsngerman{%
10712 \renewcommand*{\glossaryname}{Glossar}%
10713 \renewcommand*{\acronymname}{Akronyme}%
10714 \renewcommand*{\entryname}{Bezeichnung}%
10715 \renewcommand*{\descriptionname}{Beschreibung}%
10716 \renewcommand*{\symbolname}{Symbol}%
10717 \renewcommand*{\pagelistname}{Seiten}%
10718 \renewcommand*{\glssymbolsgroupname}{Symbole}%
10719 \renewcommand*{\glsnumbersgroupname}{Zahlen}}
10720 }

```

Italian:

```

10721 \@ifundefined{captionsitalian}{}{%
10722 \addto\captionsitalian{%
10723 \renewcommand*{\glossaryname}{Glossario}%
10724 \renewcommand*{\acronymname}{Acronimi}%
10725 \renewcommand*{\entryname}{Nomenclatura}%
10726 \renewcommand*{\descriptionname}{Descrizione}%
10727 \renewcommand*{\symbolname}{Simbolo}%
10728 \renewcommand*{\pagelistname}{Elenco delle pagine}%
10729 \renewcommand*{\glssymbolsgroupname}{Simboli}%
10730 \renewcommand*{\glsnumbersgroupname}{Numeri}}
10731 }

```

Dutch:

```

10732 \@ifundefined{captionsdutch}{}{%
10733 \addto\captionsdutch{%
10734 \renewcommand*{\glossaryname}{Woordenlijst}%
10735 \renewcommand*{\acronymname}{Acroniemen}%
10736 \renewcommand*{\entryname}{Benaming}%
10737 \renewcommand*{\descriptionname}{Beschrijving}%
10738 \renewcommand*{\symbolname}{Symbool}%
10739 \renewcommand*{\pagelistname}{Pagina's}%
10740 \renewcommand*{\glssymbolsgroupname}{Symbolen}%
10741 \renewcommand*{\glsnumbersgroupname}{Cijfers}}
10742 }

```

Spanish:

```

10743 \@ifundefined{captionsspanish}{}{%
10744 \addto\captionsspanish{%
10745 \renewcommand*{\glossaryname}{Glosario}%
10746 \renewcommand*{\acronymname}{Siglas}%

```

```

10747 \renewcommand*{\entryname}{Entrada}%
10748 \renewcommand*{\descriptionname}{Descripci'on}%
10749 \renewcommand*{\symbolname}{S'\i mbolo}%
10750 \renewcommand*{\pagelistname}{Lista de p'aginas}%
10751 \renewcommand*{\glssymbolsgroupname}{S'\i mbolos}%
10752 \renewcommand*{\glsnumbersgroupname}{N'umeros}}
10753 }

```

French:

```

10754 \@ifundefined{captionsfrench}{}{%
10755 \addto\captionsfrench{%
10756 \renewcommand*{\glossaryname}{Glossaire}%
10757 \renewcommand*{\acronymname}{Acronymes}%
10758 \renewcommand*{\entryname}{Terme}%
10759 \renewcommand*{\descriptionname}{Description}%
10760 \renewcommand*{\symbolname}{Symbole}%
10761 \renewcommand*{\pagelistname}{Pages}%
10762 \renewcommand*{\glssymbolsgroupname}{Symboles}%
10763 \renewcommand*{\glsnumbersgroupname}{Nombres}}
10764 }
10765 \@ifundefined{captionsfrenchb}{}{%
10766 \addto\captionsfrenchb{%
10767 \renewcommand*{\glossaryname}{Glossaire}%
10768 \renewcommand*{\acronymname}{Acronymes}%
10769 \renewcommand*{\entryname}{Terme}%
10770 \renewcommand*{\descriptionname}{Description}%
10771 \renewcommand*{\symbolname}{Symbole}%
10772 \renewcommand*{\pagelistname}{Pages}%
10773 \renewcommand*{\glssymbolsgroupname}{Symboles}%
10774 \renewcommand*{\glsnumbersgroupname}{Nombres}}
10775 }
10776 \@ifundefined{captionsfrancais}{}{%
10777 \addto\captionsfrancais{%
10778 \renewcommand*{\glossaryname}{Glossaire}%
10779 \renewcommand*{\acronymname}{Acronymes}%
10780 \renewcommand*{\entryname}{Terme}%
10781 \renewcommand*{\descriptionname}{Description}%
10782 \renewcommand*{\symbolname}{Symbole}%
10783 \renewcommand*{\pagelistname}{Pages}%
10784 \renewcommand*{\glssymbolsgroupname}{Symboles}%
10785 \renewcommand*{\glsnumbersgroupname}{Nombres}}
10786 }

```

Danish:

```

10787 \@ifundefined{captionsdanish}{}{%
10788 \addto\captionsdanish{%
10789 \renewcommand*{\glossaryname}{Ordliste}%
10790 \renewcommand*{\acronymname}{Akronymer}%
10791 \renewcommand*{\entryname}{Symbolforklaring}%
10792 \renewcommand*{\descriptionname}{Beskrivelse}%

```

```

10793 \renewcommand*{\symbolname}{Symbol}%
10794 \renewcommand*{\pagelistname}{Side}%
10795 \renewcommand*{\glssymbolsgroupname}{Symboler}%
10796 \renewcommand*{\glsnumbersgroupname}{Tal}}
10797 }

```

Irish:

```

10798 \@ifundefined{captionsirish}{}{%
10799 \addto\captionsirish{%
10800 \renewcommand*{\glossaryname}{Gluais}%
10801 \renewcommand*{\acronymname}{Acrainmneacha}%

```

wasn't sure whether to go for Nóta (Note), Ciall ('Meaning', 'sense') or Bri ('Meaning'). In the end I chose Ciall.

```

10802 \renewcommand*{\entryname}{Ciall}%
10803 \renewcommand*{\descriptionname}{Tuairisc}%

```

Again, not sure whether to use Comhartha/Comharthaí or Siombail/Siombaile, so have chosen the former.

```

10804 \renewcommand*{\symbolname}{Comhartha}%
10805 \renewcommand*{\glssymbolsgroupname}{Comhartha\'}{\i}}%
10806 \renewcommand*{\pagelistname}{Leathanaigh}%
10807 \renewcommand*{\glsnumbersgroupname}{Uimhreacha}}
10808 }

```

Hungarian:

```

10809 \@ifundefined{captionsmagyar}{}{%
10810 \addto\captionsmagyar{%
10811 \renewcommand*{\glossaryname}{Sz\'ojegyz\'ek}%
10812 \renewcommand*{\acronymname}{Bet\H uszavak}%
10813 \renewcommand*{\entryname}{Kifejez\'es}%
10814 \renewcommand*{\descriptionname}{Magyar\'azat}%
10815 \renewcommand*{\symbolname}{Jel\'ol\'es}%
10816 \renewcommand*{\pagelistname}{Oldalsz\'am}%
10817 \renewcommand*{\glssymbolsgroupname}{Jelek}%
10818 \renewcommand*{\glsnumbersgroupname}{Sz\'amjegyek}%
10819 }
10820 }
10821 \@ifundefined{captionshungarian}{}{%
10822 \addto\captionshungarian{%
10823 \renewcommand*{\glossaryname}{Sz\'ojegyz\'ek}%
10824 \renewcommand*{\acronymname}{Bet\H uszavak}%
10825 \renewcommand*{\entryname}{Kifejez\'es}%
10826 \renewcommand*{\descriptionname}{Magyar\'azat}%
10827 \renewcommand*{\symbolname}{Jel\'ol\'es}%
10828 \renewcommand*{\pagelistname}{Oldalsz\'am}%
10829 \renewcommand*{\glssymbolsgroupname}{Jelek}%
10830 \renewcommand*{\glsnumbersgroupname}{Sz\'amjegyek}%
10831 }
10832 }

```

## Polish

```

10833 \@ifundefined{captionspolish}{}{%
10834   \addto\captionspolish{%
10835     \renewcommand*{\glossaryname}{S{\l}ownik termin\'}ow}%
10836     \renewcommand*{\acronymname}{Skr\'}ot}%
10837     \renewcommand*{\entryname}{Termin}%
10838     \renewcommand*{\descriptionname}{Opis}%
10839     \renewcommand*{\symbolname}{Symbol}%
10840     \renewcommand*{\pagelistname}{Strony}%
10841     \renewcommand*{\glssymbolsgroupname}{Symbole}%
10842     \renewcommand*{\glnumbersgroupname}{Liczby}}
10843 }

```

## Brazilian

```

10844 \@ifundefined{captionsbrazil}{}{%
10845   \addto\captionsbrazil{%
10846     \renewcommand*{\glossaryname}{Gloss\'}ario}%
10847     \renewcommand*{\acronymname}{Siglas}%
10848     \renewcommand*{\entryname}{Nota\ c\ ~ao}%
10849     \renewcommand*{\descriptionname}{Descri\ c\ ~ao}%
10850     \renewcommand*{\symbolname}{S\'}imbolo}%
10851     \renewcommand*{\pagelistname}{Lista de P\'}aginas}%
10852     \renewcommand*{\glssymbolsgroupname}{S\'}imbolos}%
10853     \renewcommand*{\glnumbersgroupname}{N\'}umeros}%
10854   }%
10855 }

```

## 7.2 Polyglossia Captions

```

10856 \NeedsTeXFormat{LaTeX2e}
10857 \ProvidesPackage{glossaries-polyglossia}[2013/11/14 v4.0 (NLCT)]

```

### English:

```

10858 \@ifundefined{captionseenglish}{}{%
10859   \expandafter\toks@\expandafter{\captionseenglish
10860     \renewcommand*{\glossaryname}{\textenglish{Glossary}}}%
10861     \renewcommand*{\acronymname}{\textenglish{Acronyms}}}%
10862     \renewcommand*{\entryname}{\textenglish{Notation}}}%
10863     \renewcommand*{\descriptionname}{\textenglish{Description}}}%
10864     \renewcommand*{\symbolname}{\textenglish{Symbol}}}%
10865     \renewcommand*{\pagelistname}{\textenglish{Page List}}}%
10866     \renewcommand*{\glssymbolsgroupname}{\textenglish{Symbols}}}%
10867     \renewcommand*{\glnumbersgroupname}{\textenglish{Numbers}}}%
10868   }%
10869   \edef\captionseenglish{\the\toks@}%
10870 }

```

### German:

```

10871 \@ifundefined{captionsgerman}{}{%
10872   \expandafter\toks@\expandafter{\captionsgerman
10873     \renewcommand*{\glossaryname}{\textgerman{Glossar}}}%

```

```

10874 \renewcommand*{\acronymname}{\textgerman{Akronyme}}%
10875 \renewcommand*{\entryname}{\textgerman{Bezeichnung}}%
10876 \renewcommand*{\descriptionname}{\textgerman{Beschreibung}}%
10877 \renewcommand*{\symbolname}{\textgerman{Symbol}}%
10878 \renewcommand*{\pagelistname}{\textgerman{Seiten}}%
10879 \renewcommand*{\glssymbolsgroupname}{\textgerman{Symbole}}%
10880 \renewcommand*{\glsnumbersgroupname}{\textgerman{Zahlen}}%
10881 }%
10882 \edef\captionsgerman{\the\toks@}%
10883 }

Italian:
10884 \@ifundefined{captionsspanish}{\textspanish{Glosario}}%
10885 \expandafter\toks@\expandafter{\captionsspanish
10886 \renewcommand*{\glossaryname}{\textspanish{Glosario}}%
10887 \renewcommand*{\acronymname}{\textspanish{Siglas}}%
10888 \renewcommand*{\entryname}{\textspanish{Entrada}}%
10889 \renewcommand*{\descriptionname}{\textspanish{Descripci' on}}%
10890 \renewcommand*{\symbolname}{\textspanish{S' \i mbolo}}%
10891 \renewcommand*{\pagelistname}{\textspanish{Lista de p' aginas}}%
10892 \renewcommand*{\glssymbolsgroupname}{\textspanish{S' \i mbolos}}%
10893 \renewcommand*{\glsnumbersgroupname}{\textspanish{Numeri}}%
10894 }%
10895 \edef\captionsspanish{\the\toks@}%
10896 }

Dutch:
10897 \@ifundefined{captionsspanish}{\textspanish{Glosario}}%
10898 \expandafter\toks@\expandafter{\captionsspanish
10899 \renewcommand*{\glossaryname}{\textspanish{Glosario}}%
10900 \renewcommand*{\acronymname}{\textspanish{Siglas}}%
10901 \renewcommand*{\entryname}{\textspanish{Entrada}}%
10902 \renewcommand*{\descriptionname}{\textspanish{Descripci' on}}%
10903 \renewcommand*{\symbolname}{\textspanish{S' \i mbolo}}%
10904 \renewcommand*{\pagelistname}{\textspanish{Lista de p' aginas}}%
10905 \renewcommand*{\glssymbolsgroupname}{\textspanish{S' \i mbolos}}%
10906 \renewcommand*{\glsnumbersgroupname}{\textspanish{Numeri}}%
10907 }%
10908 \edef\captionsspanish{\the\toks@}%
10909 }

Spanish:
10910 \@ifundefined{captionsspanish}{\textspanish{Glosario}}%
10911 \expandafter\toks@\expandafter{\captionsspanish
10912 \renewcommand*{\glossaryname}{\textspanish{Glosario}}%
10913 \renewcommand*{\acronymname}{\textspanish{Siglas}}%
10914 \renewcommand*{\entryname}{\textspanish{Entrada}}%
10915 \renewcommand*{\descriptionname}{\textspanish{Descripci' on}}%
10916 \renewcommand*{\symbolname}{\textspanish{S' \i mbolo}}%
10917 \renewcommand*{\pagelistname}{\textspanish{Lista de p' aginas}}%
10918 \renewcommand*{\glssymbolsgroupname}{\textspanish{S' \i mbolos}}%

```

```

10919 \renewcommand*{\glnumbersgroupname}{\textspanish{N\'umeros}}%
10920 }%
10921 \edef\captionsspanish{\the\toks@}%
10922 }

```

French:

```

10923 \@ifundefined{captionsfrench}{}{%
10924 \expandafter\toks@\expandafter{\captionsfrench
10925 \renewcommand*{\glossaryname}{\textfrench{Glossaire}}%
10926 \renewcommand*{\acronymname}{\textfrench{Acronymes}}%
10927 \renewcommand*{\entryname}{\textfrench{Terme}}%
10928 \renewcommand*{\descriptionname}{\textfrench{Description}}%
10929 \renewcommand*{\symbolname}{\textfrench{Symbole}}%
10930 \renewcommand*{\pagelistname}{\textfrench{Pages}}%
10931 \renewcommand*{\glssymbolsgroupname}{\textfrench{Symboles}}%
10932 \renewcommand*{\glnumbersgroupname}{\textfrench{Nombres}}%
10933 }%
10934 \edef\captionsfrench{\the\toks@}%
10935 }

```

Danish:

```

10936 \@ifundefined{captionsdanish}{}{%
10937 \expandafter\toks@\expandafter{\captionsdanish
10938 \renewcommand*{\glossaryname}{\textdanish{Ordliste}}%
10939 \renewcommand*{\acronymname}{\textdanish{Akronymer}}%
10940 \renewcommand*{\entryname}{\textdanish{Symbolforklaring}}%
10941 \renewcommand*{\descriptionname}{\textdanish{Beskrivelse}}%
10942 \renewcommand*{\symbolname}{\textdanish{Symbol}}%
10943 \renewcommand*{\pagelistname}{\textdanish{Side}}%
10944 \renewcommand*{\glssymbolsgroupname}{\textdanish{Symboler}}%
10945 \renewcommand*{\glnumbersgroupname}{\textdanish{Tal}}%
10946 }%
10947 \edef\captionsdanish{\the\toks@}%
10948 }

```

Irish:

```

10949 \@ifundefined{captionsirish}{}{%
10950 \expandafter\toks@\expandafter{\captionsirish
10951 \renewcommand*{\glossaryname}{\textirish{Gluais}}%
10952 \renewcommand*{\acronymname}{\textirish{Acrainmneacha}}%
10953 \renewcommand*{\entryname}{\textirish{Ciall}}%
10954 \renewcommand*{\descriptionname}{\textirish{Tuaisc}}%
10955 \renewcommand*{\symbolname}{\textirish{Comhartha}}%
10956 \renewcommand*{\glssymbolsgroupname}{\textirish{Comhartha\'{\i}}}%
10957 \renewcommand*{\pagelistname}{\textirish{Leathanaigh}}%
10958 \renewcommand*{\glnumbersgroupname}{\textirish{Uimhreacha}}%
10959 }%
10960 \edef\captionsirish{\the\toks@}%
10961 }

```

Hungarian:

```

10962 \@ifundefined{captionsmagyar}{}{%

```

```

10963 \expandafter\toks@\expandafter{\captionsmagyar
10964 \renewcommand*{\glossaryname}{\textmagyar{Sz\'ojegyz\'ek}}}%
10965 \renewcommand*{\acronymname}{\textmagyar{Bet\H uszavak}}}%
10966 \renewcommand*{\entryname}{\textmagyar{Kifejez\'es}}}%
10967 \renewcommand*{\descriptionname}{\textmagyar{Magyar\'azat}}}%
10968 \renewcommand*{\symbolname}{\textmagyar{Jel\'ol\'es}}}%
10969 \renewcommand*{\pagelistname}{\textmagyar{Oldalsz\'am}}}%
10970 \renewcommand*{\glssymbolsgroupname}{\textmagyar{Jelek}}}%
10971 \renewcommand*{\glsnumbersgroupname}{\textmagyar{Sz\'amjegyek}}}%
10972 }%
10973 \edef\captionsmagyar{\the\toks@}%
10974 }

Polish
10975 \@ifundefined{captionspolish}{}{%
10976 \expandafter\toks@\expandafter{\captionspolish
10977 \renewcommand*{\glossaryname}{\textpolish{S\lownik termin\'ow}}}%
10978 \renewcommand*{\acronymname}{\textpolish{Skr\'ot}}}%
10979 \renewcommand*{\entryname}{\textpolish{Termin}}}%
10980 \renewcommand*{\descriptionname}{\textpolish{Opis}}}%
10981 \renewcommand*{\symbolname}{\textpolish{Symbol}}}%
10982 \renewcommand*{\pagelistname}{\textpolish{Strony}}}%
10983 \renewcommand*{\glssymbolsgroupname}{\textpolish{Symbole}}}%
10984 \renewcommand*{\glsnumbersgroupname}{\textpolish{Liczby}}}%
10985 }%
10986 \edef\captionspolish{\the\toks@}%
10987 }

Portugues
10988 \@ifundefined{captionsportuges}{}{%
10989 \expandafter\toks@\expandafter{\captionsportuges
10990 \renewcommand*{\glossaryname}{\textportuges{Gloss\'ario}}}%
10991 \renewcommand*{\acronymname}{\textportuges{Siglas}}}%
10992 \renewcommand*{\entryname}{\textportuges{Nota\c c\~ao}}}%
10993 \renewcommand*{\descriptionname}{\textportuges{Descri\c c\~ao}}}%
10994 \renewcommand*{\symbolname}{\textportuges{S\'imbolo}}}%
10995 \renewcommand*{\pagelistname}{\textportuges{Lista de P\'aginas}}}%
10996 \renewcommand*{\glssymbolsgroupname}{\textportuges{S\'imbolos}}}%
10997 \renewcommand*{\glsnumbersgroupname}{\textportuges{N\'umeros}}}%
10998 }%
10999 \edef\captionsportuges{\the\toks@}%
11000 }

```

### 7.3 Brazilian Dictionary

This is a dictionary file provided by Thiago de Melo for use with the package.

```
11001 \ProvidesDictionary{glossaries-dictionary}{Brazilian}
```

Provide Brazilian translations:

```

11002 \providetranslation{Glossary}{Gloss\'ario}
11003 \providetranslation{Acronyms}{Siglas}
11004 \providetranslation{Notation (glossaries)}{Nota\c c\~ao}

```

```

11005 \providetranslation{Description (glossaries)}{Descri\c c\~ao}
11006 \providetranslation{Symbol (glossaries)}{S\`imbolo}
11007 \providetranslation{Page List (glossaries)}{Lista de P\`aginas}
11008 \providetranslation{Symbols (glossaries)}{S\`imbolos}
11009 \providetranslation{Numbers (glossaries)}{N\`umeros}

```

## 7.4 Danish Dictionary

This is a dictionary file provided for use with the package.

```

11010 \ProvidesDictionary{glossaries-dictionary}{Danish}

```

Provide Danish translations:

```

11011 \providetranslation{Glossary}{Ordliste}
11012 \providetranslation{Acronyms}{Akronymer}
11013 \providetranslation{Notation (glossaries)}{Symbolforklaring}
11014 \providetranslation{Description (glossaries)}{Beskrivelse}
11015 \providetranslation{Symbol (glossaries)}{Symbol}
11016 \providetranslation{Page List (glossaries)}{Side}
11017 \providetranslation{Symbols (glossaries)}{Symboler}
11018 \providetranslation{Numbers (glossaries)}{Tal}

```

## 7.5 Dutch Dictionary

This is a dictionary file provided for use with the package.

```

11019 \ProvidesDictionary{glossaries-dictionary}{Dutch}

```

Provide Dutch translations:

```

11020 \providetranslation{Glossary}{Woordenlijst}
11021 \providetranslation{Acronyms}{Acroniemen}
11022 \providetranslation{Notation (glossaries)}{Benaming}
11023 \providetranslation{Description (glossaries)}{Beschrijving}
11024 \providetranslation{Symbol (glossaries)}{Symbool}
11025 \providetranslation{Page List (glossaries)}{Pagina's}
11026 \providetranslation{Symbols (glossaries)}{Symbolen}
11027 \providetranslation{Numbers (glossaries)}{Cijfers}

```

## 7.6 English Dictionary

This is a dictionary file provided for use with the package.

```

11028 \ProvidesDictionary{glossaries-dictionary}{English}

```

Provide English translations:

```

11029 \providetranslation{Glossary}{Glossary}
11030 \providetranslation{Acronyms}{Acronyms}
11031 \providetranslation{Notation (glossaries)}{Notation}
11032 \providetranslation{Description (glossaries)}{Description}
11033 \providetranslation{Symbol (glossaries)}{Symbol}
11034 \providetranslation{Page List (glossaries)}{Page List}
11035 \providetranslation{Symbols (glossaries)}{Symbols}
11036 \providetranslation{Numbers (glossaries)}{Numbers}

```



## 7.7 French Dictionary

This is a dictionary file provided for use with the package.

```
11037 \ProvidesDictionary{glossaries-dictionary}{French}
```

Provide French translations:

```
11038 \providetranslation{Glossary}{Glossaire}
11039 \providetranslation{Acronyms}{Acronymes}
11040 \providetranslation{Notation (glossaries)}{Terme}
11041 \providetranslation{Description (glossaries)}{Description}
11042 \providetranslation{Symbol (glossaries)}{Symbole}
11043 \providetranslation{Page List (glossaries)}{Pages}
11044 \providetranslation{Symbols (glossaries)}{Symboles}
11045 \providetranslation{Numbers (glossaries)}{Nombres}
```

## 7.8 German Dictionary

This is a dictionary file provided for use with the package.

```
11046 \ProvidesDictionary{glossaries-dictionary}{German}
```

Provide German translations (quite a few variations were suggested for German; I settled on the following):

```
11047 \providetranslation{Glossary}{Glossar}
11048 \providetranslation{Acronyms}{Akronyme}
11049 \providetranslation{Notation (glossaries)}{Bezeichnung}
11050 \providetranslation{Description (glossaries)}{Beschreibung}
11051 \providetranslation{Symbol (glossaries)}{Symbol}
11052 \providetranslation{Page List (glossaries)}{Seiten}
11053 \providetranslation{Symbols (glossaries)}{Symbole}
11054 \providetranslation{Numbers (glossaries)}{Zahlen}
```

## 7.9 Irish Dictionary

This is a dictionary file provided for use with the package.

```
11055 \ProvidesDictionary{glossaries-dictionary}{Irish}
```

Provide Irish translations:

```
11056 \providetranslation{Glossary}{Gluais}
11057 \providetranslation{Acronyms}{Acrainmneacha}
11058 \providetranslation{Notation (glossaries)}{Ciall}
11059 \providetranslation{Description (glossaries)}{Tuairisc}
11060 \providetranslation{Symbol (glossaries)}{Comhartha}
11061 \providetranslation{Page List (glossaries)}{Leathanaigh}
11062 \providetranslation{Symbols (glossaries)}{Comhartha'\i}
11063 \providetranslation{Numbers (glossaries)}{Uimhreacha}
```

## 7.10 Italian Dictionary

This is a dictionary file provided for use with the package.

```
11064 \ProvidesDictionary{glossaries-dictionary}{Italian}
```

Provide Italian translations:

```
11065 \providetranslation{Glossary}{Glossario}
11066 \providetranslation{Acronyms}{Acronimi}
11067 \providetranslation{Notation (glossaries)}{Nomenclatura}
11068 \providetranslation{Description (glossaries)}{Descrizione}
11069 \providetranslation{Symbol (glossaries)}{Simbolo}
11070 \providetranslation{Page List (glossaries)}{Elenco delle pagine}
11071 \providetranslation{Symbols (glossaries)}{Simboli}
11072 \providetranslation{Numbers (glossaries)}{Numeri}
```

## 7.11 Magyar Dictionary

This is a dictionary file provided for use with the package.

```
11073 \ProvidesDictionary{glossaries-dictionary}{Magyar}
```

Provide translations:

```
11074 \providetranslation{Glossary}{Sz\’ojegyz\’ek}
11075 \providetranslation{Acronyms}{Bet\H uszavak}
11076 \providetranslation{Notation (glossaries)}{Kifejez\’es}
11077 \providetranslation{Description (glossaries)}{Magyar\’azat}
11078 \providetranslation{Symbol (glossaries)}{Jel\’ol\’es}
11079 \providetranslation{Page List (glossaries)}{Oldalsz\’am}
11080 \providetranslation{Symbols (glossaries)}{Jelek}
11081 \providetranslation{Numbers (glossaries)}{Sz\’amjegyek}
```

## 7.12 Polish Dictionary

This is a dictionary file provided for use with the package.

```
11082 \ProvidesDictionary{glossaries-dictionary}{Polish}
```

Provide Polish translations:

```
11083 \providetranslation{Glossary}{S{\l}ownik termin\’ow}
11084 \providetranslation{Acronyms}{Skr\’ot}
11085 \providetranslation{Notation (glossaries)}{Termin}
11086 \providetranslation{Description (glossaries)}{Opis}
11087 \providetranslation{Symbol (glossaries)}{Symbol}
11088 \providetranslation{Page List (glossaries)}{Strony}
11089 \providetranslation{Symbols (glossaries)}{Symbole}
11090 \providetranslation{Numbers (glossaries)}{Liczby}
```

## 7.13 Serbian Dictionary

This dictionary was provided by Zoran Filipovic.

```
11091 \ProvidesDictionary{glossaries-dictionary}{Serbian}
11092 \providetranslation{Glossary}{Mali re\’v cnik}
11093 \providetranslation{Acronyms}{Skra\’ cenice}
11094 \providetranslation{Notation (glossaries)}{Oznaka}
11095 \providetranslation{Description (glossaries)}{Opis}
11096 \providetranslation{Symbol (glossaries)}{Simbol}
```

```

11097 \providetranslation{Page List (glossaries)}{Stranica}
11098 \providetranslation{Symbols (glossaries)}{Simboli}
11099 \providetranslation{Numbers (glossaries)}{Brojevi}

```

## 7.14 Spanish Dictionary

This is a dictionary file provided for use with the package.

```
11100 \ProvidesDictionary{glossaries-dictionary}{Spanish}
```

Provide Spanish translations:

```

11101 \providetranslation{Glossary}{Glosario}
11102 \providetranslation{Acronyms}{Siglas}
11103 \providetranslation{Notation (glossaries)}{Entrada}
11104 \providetranslation{Description (glossaries)}{Descripci\on}
11105 \providetranslation{Symbol (glossaries)}{S\'\i mbolo}
11106 \providetranslation{Page List (glossaries)}{Lista de p\ aginas}
11107 \providetranslation{Symbols (glossaries)}{S\'\i mbolos}
11108 \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 ..... 68
	<code>\writeist</code> : Added spaces after <code>\delimN</code> and <code>\delimR</code> in ist file ..... 143	<code>\glsmakefirstuc</code> : new ..... 241
1.03	<code>\makefirstuc</code> : changed 'pro- tected@edef to 'def ..... 240	1.06 General: now requires etoolbox . 240
1.04	General: Added <code>\glstextformat</code> 78	<code>\capitalisewords</code> : new ..... 241
1.05	<code>\glossarysection</code> : added <code>\@mkboth to \glossarysection</code> ..... 35	<code>\xcapitalisewords</code> : new .... 242
	<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 ..... 141

- 1.08
- General: Added babel support ... 29
  - \capitalisewords: made robust ..... 241
  - listgroup: changed listgroup style to use \glsgetgrouptitle ..... 247
  - altlistgroup: changed altlistgroup style to use \glsgetgrouptitle ..... 248
  - \makefirstuc: made robust ... 240
- 1.1
- \@glossarysection: numbered sections and auto label added 36
  - \@gls@tmpb: changed \toksdef to \newtoks ..... 94
  - \@gls@toc: numberline added .. 37
  - \@p@glossarysection: numbered sections and auto label added ..... 36
  - General: Added support for translator package ..... 30
  - amsgen now loaded (\new@ifnextchar needed) ..... 4
  - translate: translate option added ..... 21
  - \setglossarysection: new ... 36
  - numberedsection: numbered-section package option added . 6
  - numberline: numberline option added ..... 5
- 1.12
- \@GLSpl: now uses \glsentrydescplural and \glsentrysymbolplural instead of \glsentrydesc and \glsentrysymbol .... 108
  - \@GLspl@: now uses \glsentrydescplural and \glsentrysymbolplural instead of \glsentrydesc and \glsentrysymbol .... 106
  - General: added check for \hypertarget separate to \hyperlink (memoir defines \hyperlink but not \hypertarget) ..... 100
  - descriptionplural: new ..... 55
  - \gls@defglossaryentry: Changed default first plural to be first key with s appended (was text key with s appended) 68
  - descriptionplural support added ..... 68
  - symbolplural support added .. 68
  - \Glsentrydescplural: New .. 135
  - \glsentrydescplural: New .. 135
  - \Glsentrysymbolplural: New 136
  - \glsentrysymbolplural: New 136
  - \SetDescriptionFootnoteAcronymStyle: Added \protect before \footnote and \glslink . 213
  - \SetFootnoteAcronymStyle: Added \protect before \footnote and \glslink . 220
  - symbolplural: new ..... 57
- 1.13
- General: Add Polish support 348, 351
  - fixed bug that ignored 3rd parameter ..... 110–119
  - \ACRfullpl: new ..... 195
  - \Acrfullpl: new ..... 195
  - \acrfullpl: new ..... 194
  - \acrpluralsuffix: New ..... 192
  - \gls@defglossaryentry: Changed default first value .. 68
  - Changed default firstplural value ..... 68
  - Removed restriction on only using \newglossaryentry in the preamble ..... 73
  - \newacronym: Removed restriction on only using \newacronym in the preamble 192
- 1.14
- \@gls@hypergroup: new ..... 243
  - General: added nonnumberlist key to \printglossary ..... 178
  - added numberedsection key to \printglossary ..... 177
  - \firstacronymfont: new .... 196
  - \glsautoprefix: new ..... 6
  - \glsnavhyperlink: changed 'edef to 'protected@edef ... 242
  - \glsnavhypertarget: added write to aux file ..... 243
  - \glsnavigation: changed to only use labels for groups that are present ..... 244

1.15	1.17
\@gls@link: added \glslabel . 90	\@@do@wrglossary: new ..... 159
General: Added \glssettoctitle	\@do@seeglossary: new ..... 161
..... 30	\@glo@storeentry: new ..... 74
\gls@defglossaryentry: check	\@glossary: changed defini-
for \@glo@first in descrip-	tion to use \index instead of
tion ..... 72	\@index ..... 157
check for \@glo@text in sym-	\@glsdefaultplural: new .... 59
bol ..... 72	\@glsdefaultsort: new ..... 60
\gls@hypergroup: new . 243	\@gls@hypernumber: new ..... 189
\glsnavhypertarget: added	\@glsnoname: new ..... 59
check if rerun required .... 243	\@glsnonextpages: new ..... 178
\glssettoctitle: new ..... 29	\@wrglossary: modified to allow
\printglossary: changed the	for xindy support ..... 158
way the TOC title is set .... 164	General: added Brazilian dictio-
1.16	nary ..... 351
\@GLS@: Test glossary type is	Added Brazilian support .... 348
\acronymtype in addition to	added xindy support ..... 23
checking if footnote option has	parent: new ..... 57
been used ..... 104	see: new ..... 57
\@GLSpl: Test glossary type is	\gls@defglossaryentry: added
\acronymtype in addition to	nonumberlist key ..... 68
checking if footnote option has	added parent key ..... 68
been used ..... 108	added see key ..... 68
\@Gls@: Test glossary type is	Stored main part of entry format
\acronymtype in addition to	when entry is defined ..... 73
checking if footnote option has	\gls@suffixF: new ..... 33
been used ..... 103	\gls@suffixFF: new ..... 33
\@Glspl@: Test glossary type is	\gls@hyperlink: new ..... 141
\acronymtype in addition to	\gls@hypernumber: modified to
checking if footnote option has	allow material to be attached
been used ..... 106	to location ..... 188
\@Gls@: Test glossary type is	\glsnavhyperlink: replaced 'hy-
\acronymtype in addition to	perlink to '@glslink ..... 242
checking if footnote option has	\glsnavhypertarget: replaced
been used ..... 102	'hypertarget to '@glstarget . 243
\@Glsdisp: Test glossary type is	\glssee: new ..... 162
\acronymtype in addition to	\glsseeformat: new ..... 162
checking if footnote option has	\glsSetSuffixF: new ..... 33
been used ..... 109	\glsSetSuffixFF: new ..... 33
\@Glspl@: Test glossary type is	\ifglsxindy: new ..... 23
\acronymtype in addition to	\istfilename: added xindy sup-
checking if footnote option has	port ..... 32
been used ..... 105	\newglossarystyle: made
\@glstarget: raised the hyper-	\newglossarystyle long . 187
target so the target text doesn't	\nopostdesc: new ..... 31
scroll off the top of the page 100	nonumberlist: new ..... 58
\gls@defglossaryentry:	
Changed def to let ..... 68	

\printglossary: added check to determine if \printglossary is already defined ..... 164 added print language to aux file 164 order: order package option added ..... 23 \writeist: added xindy support 143	2.01	\@gls@link: moved \@do@wrglossary before term is displayed to pre- vent unwanted whatsit ..... 90 \forallglossaries: replaced \ifthenelse with \ifx .... 47 \forglentries: replaced \ifthenelse with \ifx .... 47 \glsgldefmain: new ..... 12 \glsgldescwidth: changed \linewidth to \hspace . 250, 265 \glsglslstdottedwidth: changed \linewidth to \hspace .... 249 \glsglspagelistwidth: changed \linewidth to \hspace . 250, 265 nomain: added nomain package option ..... 13 \writeist: removed item_02 - no such makeindex key ..... 148
1.18 \@gls@loadlist: new ..... 8 \@gls@loadlong: new ..... 8 \@gls@loadsuper: new ..... 8 \@gls@loadtree: new ..... 8 \glsgldefglossaryentry: Changed default value of sort to \@glsgldefaultsort ..... 68 moved sort sanitization to \newglossaryentry ..... 72 \glsglstartarget: new ..... 182 \oldacronym: new ..... 191 nolist: new ..... 8 nolong: new ..... 8 sort: moved sanitization to \newglossaryentry ..... 56 nostyles: new ..... 8 nosuper: new ..... 8 notree: new ..... 8	2.02	\@printglossary: suppressed warning globally rather than locally ..... 166 General: Changed Brazil to Brazil- ian ..... 351 false will prevent automatic loading of translator package 27 \glossarysection: changed \@mkboth to \glossarymark 35 \glsglsglossarymark: New ..... 35
1.19 \glsglclearpage: new ..... 37 \glsgldisp: new ..... 108 \SetDescriptionAcronymStyle: changed \acronymfont to use \textsmaller instead of \smaller ..... 218 \SetDescriptionFootnoteAcronymStyle: changed \acronymfont to use \textsmaller instead of \smaller ..... 213 \SetFootnoteAcronymStyle: changed \acronymfont to use \textsmaller instead of \smaller ..... 220 \SetSmallAcronymStyle: changed \acronymfont to use \textsmaller instead of \smaller ..... 223	2.03	\@GLS@: Added check for hyper- first ..... 104 \@GLS@: Added check for hyper- first ..... 108 \@GLS@: Added check for hyper- first ..... 103 \@GLS@: Added check for hyper- first ..... 106 \@gls@: Added check for hyper- first ..... 102 \@gls@link: new ..... 89 \@gls@link: added \leavevmode ..... 90 Moved entry existence check to avoid duplicate code ..... 90 \@glsgldisp: Added check for hy- perfirst ..... 109
1.2 General: fixed bug in ngerman captions ..... 345		

\@glsp1@: Added check for hyper-first .....	105	\glentryuseriv: new .....	138
\glsglossarymark: Added check to see if it's already defined ..	35	\Glsentryuseriv: new .....	138
hyperfirst: new .....	22	\glentryuseriv: new .....	138
2.04		\glentryuseriv: new .....	138
\@GLS@: Changed test to check if glossary type has been identified as a list of acronyms ...	104	\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 ...	108	\SetAcronymLists: new .....	15
\@GLs@: Changed test to check if glossary type has been identified as a list of acronyms ...	103	\SetDefaultAcronymDisplayStyle: new .....	210
\@GLsp1@: Changed test to check if glossary type has been identified as a list of acronyms ..	106	\SetDefaultAcronymStyle: new .....	211
\@glossaryentryfield: new ..	73	\SetDescriptionAcronymDisplayStyle: new .....	215
\@glossarysubentryfield: new .....	73	\SetDescriptionDUAAcronymDisplayStyle: new .....	214
\@gl@: Changed test to check if glossary type has been identified as a list of acronyms ...	102	\SetDescriptionFootnoteAcronymDisplayStyle: new .....	212
\@gl@acronymlists: new .....	14	\SetDUADisplayStyle: new ..	223
\@gl@disp: Changed test to check if glossary type has been identified as a list of acronyms ..	109	\SetFootnoteAcronymDisplayStyle: new .....	218
\@glsp1@: Changed test to check if glossary type has been identified as a list of acronyms ..	105	\SetSmallAcronymDisplayStyle: new .....	220
\@newglossaryentryposthook: new .....	73	2.05	
\@newglossaryentryprehook: new .....	73	\@gl@disp: Added closing brace. Patch provided by Sergiu Dotenco .....	109
acronymlists: new .....	15	Removed spurious brace. Patch provided by Sergiu Dotenco ..	109
\DeclareAcronymList: new ...	14	\writeist: Added \string before opening and closing braces. Patch provided by Segiu Dotenco .....	148
\DefineAcronymSynonyms: new	208	2.06	
\gls@defglossaryentry: added user1-6 keys .....	68	\altnewglossary: new .....	54
\gl@add: fixed bug that ignored counter .....	141	\CustomAcronymFields: new ..	226
\Glsentryuseri: new .....	137	\CustomNewAcronymDef: new ..	226
\Glsentryuseri: new .....	137	\SetCustomDisplayStyle: new	225
\Glsentryuserii: new .....	137	\SetCustomStyle: new .....	226
\Glsentryuserii: new .....	137	2.07	
\Glsentryuseriii: new .....	138	General: glssadd format key stored in \@glsnumberformat (was mistakenly stored in \@glo@format) .....	141
\Glsentryuseriii: new .....	137	3.0	
\Glsentryuseriv: new .....	138	\@do@wrglossary: added check for hyper location prefix ...	160

modified to use new format ..	159	\acrfullpl: added starred ver-	
\@@glossarysec: replaced		sion .....	194
\@ifundefined with		\acrlinkfootnote:new ....	211
\ifcsundef .....	5	\acrnolinkfootnote:new ...	211
\@do@seeglossary: Sanitize and		\addglossarytocaptions: re-	
escape cross-referencing in-		placed \@ifundefined with	
formation .....	161	\ifcsundef .....	29
\@gls@counterwithin:new ....	9	savewrites:new .....	24
\@gls@ifinlist:new .....	38	see:added \@glo@seeautonumberlist	
\@gls@link:added \@gls@saveentrycounter.....	57	seeautonumberlist:new .....	7
.....	90	\glossarysection: replaced	
added \@gls@setsort .....	90	\@ifundefined with	
\@gls@saveentrycounter:new	91	\ifcsundef .....	35
\@gls@setupsort@def:new ...	11	\glossarystyle: replaced	
\@gls@setupsort@standard:		\@ifundefined with	
new .....	10	\ifcsundef .....	187
\@gls@setupsort@use:new ...	11	\gls@codepage: replaced	
\@gls@xdy@locationlist:new	41	\@ifundefined with	
\@glslink:replaced \@ifundefined		\ifcsundef .....	24
with \ifcsundef .....	100	\gls@defglossaryentry: added	
\@glsnextpages:new .....	179	\@gls@defsort .....	72
\@makeglossary: Added check		added short and long keys ....	69
for savewrites .....	150	replaced \@ifundefined with	
\@print@glossary: replaced		\ifcsundef .....	69
\@ifundefined with		\gls@doclearpage: replaced	
\ifcsundef .....	167	\@ifundefined with	
\@printglossary: added		\ifcsundef .....	37
\currentglossary .....	165	\glsadd:added \@gls@saveentrycounter	
added \glsnextpages .....	165	.....	142
make toctitle default to title ..	165	\GlsAddXdyCounters:new ....	38
\@set@glo@numformat: added		\glentrycounterlabel:new	181
4th argument .....	92	\glentryitem:new .....	181
\@wrglossary: modified to take		\Glentrylong:new .....	139
into account savewrites ....	158	\glentrylong:new .....	139
\@xdyattributelist:new ....	38	\Glentrylongpl:new .....	139
General: added prefix to hyperlink		\glentrylongpl:new .....	139
.....	190	\Glentryshort:new .....	138
etoolbox now loaded .....	4	\glentryshort:new .....	138
replaced \@ifundefined with		\Glentryshortpl:new ....	139
\ifcsundef .....	27, 88, 177	\glentryshortpl:new ....	139
\acrfootnote:new .....	211	\glsgetgrouptitle: re-	
\ACRfull: added starred version	194	placed \@ifundefined with	
\Acrfull: added starred version	193	\ifcsundef .....	185
\acrfull: added starred version	193	\gls glossarymark: replaced	
\ACRfullpl: added starred ver-		\@ifundefined with	
sion .....	195	\ifcsundef .....	35
\Acrfullpl: added starred ver-		\glshyperlink: changed de-	
sion .....	195	fault from \glentryname to	



<code>\glstrytext</code> .....	141	<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> .....	188	<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> .....	33	<code>\printglossary:</code>	replaced
<code>\glstrsrefentry:new</code> .....	180	<code>\@ifundefined</code>	with
<code>\glstrsresetsubentrycounter:</code>		<code>\ifcsundef</code> .....	164
<code>new</code> .....	180	<code>\SetDescriptionFootnoteAcronymDisplayStyle:</code>	
<code>\glseeitem:</code>	hyperlink uses	<code>expanded options link op-</code>	
<code>\glseeitemformat</code>	instead	<code>tions</code> .....	212
<code>of \glstryname</code> .....	163	<code>\setentrycounter:</code>	added op-
<code>\glseeitemformat:new</code> ....	163	<code>tional argument</code> .....	186
<code>\glssortnumberfmt:new</code> ....	10	<code>\showacronymlists:new</code> ....	232
<code>\glsstrstepentry:new</code> .....	180	<code>\showglocounter:new</code> .....	229
<code>\glsstrstepsubentry:new</code> ....	180	<code>\showgloDESC:new</code> .....	230
<code>\glstrsubentrycounterlabel:</code>		<code>\showgloDESCplural:new</code> ...	230
<code>new</code> .....	181	<code>\showglofirst:new</code> .....	228
<code>\glstrsubentryitem:new</code> ....	181	<code>\showglofirstpl:new</code> .....	228
<code>theglossary:replaced \@ifundefined</code>		<code>\showgloflag:new</code> .....	231
<code>with \ifcsundef</code> .....	181	<code>\showgloindex:new</code> .....	231
<code>short:new</code> .....	58	<code>\showglolevel:new</code> .....	228
<code>shortplural:new</code> .....	59	<code>\showgloNAME:new</code> .....	230
<code>\ifglossaryexists:</code>	re-	<code>\showgloparent:new</code> .....	227
<code>placed \@ifundefined with</code>		<code>\showgloplural:new</code> .....	228
<code>\ifcsundef</code> .....	47	<code>\showglosort:new</code> .....	230
<code>\ifglstryexists:</code>	re-	<code>\showglossaries:new</code> .....	232
<code>placed \@ifundefined with</code>		<code>\showglossarycounter:new</code> .	232
<code>\ifcsundef</code> .....	48	<code>\showglossaryentries:new</code> .	233
<code>\istfile: deprecated</code> .....	156	<code>\showglossaryin:new</code> .....	232
<code>glossaryentry:new</code> .....	179	<code>\showglossaryout:new</code> ....	232
<code>glossarysubentry:new</code> ....	179	<code>\showglossarytitle:new</code> ...	232
<code>\newglossary:added \@glstrsortcount</code>		<code>\showglosymbol:new</code> .....	230
<code>.....</code> .....	54	<code>\showglosymbolplural:new</code> .	231
<code>replaced \@ifundefined with</code>		<code>\showgloTEXT:new</code> .....	228
<code>\ifcsundef</code> .....	54	<code>\showgloTYPE:new</code> .....	228
<code>\newglossaryentry:</code>	replaced	<code>\showgloUSERi:new</code> .....	229
<code>\DeclareRobustCommand</code>		<code>\showgloUSERii:new</code> .....	229
<code>with \newrobustcmd</code> .....	62	<code>\showgloUSERiii:new</code> .....	229
<code>\newglossarystyle:</code>	re-	<code>\showgloUSERiv:new</code> .....	229
<code>placed \@ifundefined with</code>		<code>\showgloUSERv:new</code> .....	229
<code>\ifcsundef</code> .....	187	<code>\showgloUSERvi:new</code> .....	230
<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> .....	191	<code>open tag</code> .....	146
<code>compatible-2.07: compatible-</code>		<code>modified to support new for-</code>	
<code>2.07 option added</code> .....	25	<code>mat</code> .....	143

3.01	
\@glswritefiles: added check for empty glossaries . . . . .	156
General: made robust . . . . .	103
\ACRfull: made robust . . . . .	194
\Acrfull: made robust . . . . .	193
\acrfull: made robust . . . . .	192
\acrfullformat: removed \acronymfont as it should al- ready be set in the second ar- gument. . . . .	193
\ACRfullpl: made robust . . . .	195
\Acrfullpl: made robust . . . .	195
\acrfullpl: made robust . . . .	194
\ACRlong: made robust . . . . .	131
\Acrlong: made robust . . . . .	130
\acrlong: made robust . . . . .	130
\ACRlongpl: made robust . . . .	133
\Acrlongpl: made robust . . . .	132
\acrlongpl: made robust . . . .	132
\ACRshort: made robust . . . . .	127
\Acrshort: made robust . . . . .	126
\acrshort: made robust . . . . .	126
\ACRshortpl: made robust . . . .	129
\Acrshortpl: made robust . . . .	128
\acrshortpl: made robust . . . .	128
\Gls: made robust . . . . .	102
\glsadd: made robust . . . . .	141
\glsaddall: made robust . . . .	142
\GLSdesc: made robust . . . . .	116
\Glsdesc: made robust . . . . .	115
\glsdesc: made robust . . . . .	115
\GLSdescplural: made robust . .	117
\Glsdescplural: made robust . .	116
\glsdescplural: made robust . .	116
\glsfirst: made robust . . . . .	111
\GLSfirstplural: made robust . .	113
\Glsfirstplural: made robust . .	113
\glsfirstplural: made robust . .	113
\glslink: made robust . . . . .	89
\GLSname: made robust . . . . .	115
\Glsname: made robust . . . . .	114
\glsname: made robust . . . . .	114
\GLSpl: made robust . . . . .	107
\Glspl: made robust . . . . .	106
\glspl: made robust . . . . .	105
\GLSplural: made robust . . . .	112
\GLSsymbol: made robust . . . .	118
\Glsymbol: made robust . . . .	117
	\glsymbol: made robust . . . . 117
	\GLSsymbolplural: made robust 119
	\Glsymbolplural: made robust 118
	\glsymbolplural: made robust 118
	\Glstext: made robust . . . . . 110
	\glstext: made robust . . . . . 110
	\GLSuseri: made robust . . . . . 120
	\Glsuseri: made robust . . . . . 120
	\glsuseri: made robust . . . . . 119
	\GLSuserii: made robust . . . . 121
	\Glsuserii: made robust . . . . 121
	\glsuserii: made robust . . . . 120
	\GLSuseriii: made robust . . . . 122
	\Glsuseriii: made robust . . . . 122
	\glsuseriii: made robust . . . . 121
	\GLSuseriv: made robust . . . . 123
	\Glsuseriv: made robust . . . . 123
	\glsuseriv: made robust . . . . 122
	\GLSuserv: made robust . . . . . 124
	\Glsuserv: made robust . . . . . 124
	\glsuserv: made robust . . . . . 123
	\GLSuservi: made robust . . . . 125
	\Glsuservi: made robust . . . . 125
	\glsuservi: made robust . . . . 125
3.02	
	\@do@wrglossary: changed \@glslocref to \theglstrycounter 160
	\@do@wrglossary: changed \@do@wr@glossary to test for indexonlyfirst option; put old \@do@wr@glossary code into \@do@wrglossary . . . . . 158
	\@gls@missingnumberlist: new . . . . . 59
	\@glswritefiles: added check for existence of token in case \makeglossaries has been omitted . . . . . 156
	\@printglossary: add a way to fetch current entry label . . . 166
	\@wrglossary: added check for glossary file defined . . . . . 158
	General: added check for polyglos- sia . . . . . 27
	reversed order of package check 31

savenumberlist:new .....	7	altlongragged4col: added	
ucmark:new .....	9	check for glsnogroupskip ..	260
\gls@defglossaryentry: added		altsuperragged4col: added	
numberlist element .....	71	check for glsnogroupskip ..	277
\gls@save@numberlist:new ..	163	alttree: added check for	
\glsdisplaynumberlist:new ..	140	glsnogroupskip .....	285
\glsentrycounter: set default		index: added check for	
value .....	91	glsnogroupskip .....	279
\Glsentryfull: fixed bug (re-		nogroupskip:new .....	9
placed \glsentryshortpl		long: added check for	
with \glsentryshort) ....	139	glsnogroupskip .....	251
\glsentryfullpl: fixed bug (re-		long3col: added check for	
placed \glsentryshort with		glsnogroupskip .....	252
\glsentryshortpl) .....	139	long4col: added check for	
\glsentrynumberlist:new ..	140	glsnogroupskip .....	254
\glsmoveentry:new .....	73	longragged: added check for	
\glsnumlistlastsep:new ...	141	glsnogroupskip .....	257
\glsnumlistsep:new .....	141	longragged3col: added check	
\glsresetsubentrycounter:		for glsnogroupskip .....	258
new .....	180	nopostdot:new .....	9
\ifglshaschildren:new .....	49	tree: added check for	
\ifglshasparent:new .....	50	glsnogroupskip .....	280
\makeglossaries: added list		treenoname: added check for	
parser .....	151	glsnogroupskip .....	282
indexonlyfirst:new .....	22	super: added check for	
\renewglossarystyle:new ..	188	glsnogroupskip .....	266
\showglossaryentries: fixed		super3col: added check for	
misspelt command .....	233	glsnogroupskip .....	268
\SmallNewAcronymDef: fixed		super4col: added check for	
broken short and long plural	221	glsnogroupskip .....	269
3.03		superragged: added check for	
\@gls@sanitizesort:new ....	18	glsnogroupskip .....	273
\@gls@setupsort@standard:		superragged3col: added check	
used \@gls@sanitizesort ..	10	for glsnogroupskip .....	275
\@printglossary: allow title to		3.04	
override default toctitle ....	165	\@do@wrglossary: changed	
General: allow title to set toctitle	177	\theglsentrycounter back	
\glsinlinedescformat:new ..	246	to \@glslocref .....	160
\glsinlineemptydescformat:		\@do@wrglossary: modified to	
new .....	246	compensate for possible incor-	
\glsinlinenameformat:new ..	246	rect page number .....	159
\glsinlinepostchild:new ..	246	\@gls@escbsdq: unsani-	
\glsinlinesubdescformat:		tize \gls@numberpage,	
new .....	246	\gls@alphpage, \gls@Alphpage	
\glsinlinesubnameformat:		and \gls@romanpage .....	93
new .....	246	\@print@glossary: Moved aux	
\glspostinline: replaced “.”		write to end of document to	
with \glspostdescription	246	prevent unwanted whatsit oc-	
		curing here. ....	167

General: Added check for doc	3.07	
package .....	4	\@gls@link: fixed bug that failed
added datatool-base as a re-		to find entry in list .....
quired package .....	4	90
added local key .....	89	\glossarypreamble: modified to
\gls@Alphpage: new .....	159	work with \setglossarypreamble
\gls@alphpage: new .....	159	.....
\gls@disablepagerefexpansion:		34
new .....	158	\gls@docclearpage: added check
\gls@numberpage: new .....	159	for openright .....
\gls@protected@pagefmts:		37
new .....	158	\glspostdescription: Added
\gls@romanpage: new .....	159	spacefactor code .....
\glsdefmain: added check for		8
doc package .....	12	\GlsSetXdyCodePage: Added
\glsorg@endtheglossary: new .	5	check for fontspec .....
\glsorg@glossary: new .....	4	46
\glsorg@theglossary: new ....	5	\SetDescriptionAcronymDisplayStyle:
\glsorg@wrglossary: new ....	4	now using \glsdoparenifnotempty
altlist: replaced \newline with		.....
paragraph break .....	248	215
\PrintChanges: new .....	5	\setglossarypreamble: new ..
3.05		34
\@do@wrglossary: add Roman		3.08a
case. Fixed bugs in the else		\@glo@storeentry: no longer
statements .....	159	need to check for special char-
\@gls@link: added check for “no-		acters in any of the fields other
hypertypes” .....	90	than sort .....
\@gls@nohyperlist: new ....	15	74
mcolalttree: replaced ‘2’ with		updated for \glossentry ....
\glsmcols .....	264	74
mcolindex: replaced ‘2’ with		\@glossaryentryfield: switched
\glsmcols .....	262	to \glossentry .....
mcoltree: replaced ‘2’ with		73
\glsmcols .....	262	\@glossarysubentryfield:
mcoltreename: replaced ‘2’		switched to \subglossentry
with \glsmcols .....	263	73
\gls@protected@pagefmts:		General: added nogroupskip key
added Roman to list .....	158	to \printglossary .....
\gls@Romanpage: new .....	159	178
\GlsDeclareNoHyperList: new	15	removed definition of
\glsgetgrouplabel: fixed bug		\@glossaryentryfield ..
(typo in \equal) .....	186	325
\nopostdesc: made robust ....	31	removed definition of
nohypertypes: new .....	16	\@glossarysubentryfield
3.06		325
\@xdy@main@language: Changed		\compatibleglossentry: new
back to using \language	23	182
\findrootlanguage: Obsoleted	45	\compatiblesubglossentry:
		new .....
		183
		\glossaryentryfield: depre-
		cated .....
		184
		\Glossentrydesc: new .....
		183
		\glossentrydesc: new .....
		183
		\Glossentryname: new .....
		182
		\glossentryname: new .....
		182
		\Glossentrysymbol: new ....
		183
		\glossentrysymbol: new ....
		183
		\gls@assign@desc@field: new
		17
		\gls@assign@descplural@field:
		new .....
		17
		\gls@assign@field: new ....
		61
		\gls@ifnotmeasuring: new ...
		75
		\glsaddallunused: new ....
		142

\glsexpandfields: new .....	61	3.09a	
\glсноexpandfields: new ....	62		\@gls@assign@symbolplural@field:
\glsee: made robust .....	162		new .....
\glseeformat: made robust ..	162		\@gls@default@value: new ...
\glseeitem: made robust ....	163		\Glsentrydesc: made robust ..
\glseeelist: made robust ....	162		\Glsentrydescplural: made ro-
\ifglsdessuppressed: new ..	50		bust .....
\ifglshasdesc: new .....	50		\Glsentryfirst: made robust .
\ifglshassymbol: new .....	50		\Glsentryfirstplural: made
list: updated list style to			robust .....
use \glossentry and			\Glsentryfull: made robust ..
\subglossentry .....	247		\Glsentryfullpl: made robust
listdotted: updated listdotted			\Glsentrylong: made robust ..
style to use \glossentry and			\Glsentrylongpl: made robust
\subglossentry .....	249		\Glsentryname: made robust ..
altlist: updated altlist style			\Glsentryplural: made robust
to use \glossentry and			\Glsentryshort: made robust .
\subglossentry .....	248		\Glsentryshortpl: made robust
altlongragged4col: updated			.....
to use \glossentry and			\Glsentrysymbol: made robust
\subglossentry .....	260		\Glsentrysymbolplural: made
alttree: updated to use			robust .....
\glossentry and \subglossentry			\Glsentrytext: made robust ..
.....	283		\Glsentryuseri: made robust .
index: added paragraph break at			\Glsentryuserii: made robust
end of environment .....	278		\Glsentryuseriii: made robust
updated to use \glossentry			.....
and \subglossentry .....	278		\Glsentryuseriv: made robust
inline: updated inline style			\Glsentryuserv: made robust .
to use \glossentry and			\Glsentryuservi: made robust
\subglossentry .....	245		\glstextup: new .....
long: updated to use \glossentry			\if@gls@docloaded: Add a fix for
and \subglossentry .....	250		\RecordChanges .....
longragged: updated to			\ifglshassymbol: changed test
use \glossentry and			to check for \@gls@default@symbol
\subglossentry .....	257		.....
longragged3col: updated		3.10a	
to use \glossentry and			\@gls@keymap: new .....
\subglossentry .....	258		\@gls@provide@newglossary:
tree: updated to use \glossentry			new .....
and \subglossentry .....	280		\@gls@writedef: new .....
\setglossarystyle: new ....	187		\@gls@defaultplural: Obsolete .
\setglossentrycompatibility:			\@gls@nodesc: new .....
new .....	184		\@print@glossary: Added
superragged: updated to			providecommand code to aux
use \glossentry and			file .....
\subglossentry .....	273		\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 ..... 103
..... 68	removed <code>\makefirstuc</code> (now
new ..... 68	dealt with in <code>\glsentryfmt</code> ) 103
<code>\gls@writedefhook: new</code> ..... 67	<code>\@Glspl@:</code> add <code>\glsifplural,</code>
<code>\makeglossaries:</code> Added	<code>\glscapscase, \glscustomtext</code>
providecommand code to aux	and <code>\glsinsert</code> ..... 106
file ..... 150	change to using <code>\glsentryfmt</code>
<code>\new@glossaryentry: new</code> .... 62	style commands ..... 106
<code>\newglossary: added \@gls@provide@newglossary</code>	removed <code>\makefirstuc</code> (now
..... 53	dealt with in <code>\glsentryfmt</code> ) 106
3.11a	<code>\@acrlong:</code> added <code>\glslabel,</code>
<code>\glsifplural, \glscapscase,</code>	<code>\glsifplural, \glscapscase,</code>
<code>\glsinsert and \glscustomtext</code>	<code>\glsinsert and \glscustomtext</code>
..... 325	..... 324
<code>\@ACRshort:</code> added <code>\glslabel,</code>	<code>\@acrshort:</code> added <code>\glslabel,</code>
<code>\glsifplural, \glscapscase,</code>	<code>\glsifplural, \glscapscase,</code>
<code>\glsinsert and \glscustomtext</code>	<code>\glsinsert and \glscustomtext</code>
..... 324	..... 323
<code>\@Acrlong:</code> added <code>\glslabel,</code>	<code>\@gls@:</code> add <code>\glslabel,</code>
<code>\glsifplural, \glscapscase,</code>	<code>\glsifplural, \glscapscase,</code>
<code>\glsinsert and \glscustomtext</code>	<code>\glscustomtext</code> and
..... 324	<code>\glsinsert</code> ..... 101
<code>\@Acrshort:</code> added <code>\glslabel,</code>	change to using <code>\glsentryfmt</code>
<code>\glsifplural, \glscapscase,</code>	style commands ..... 101
<code>\glsinsert and \glscustomtext</code>	<code>\@gls@noexpand@fields:</code> Fixed
..... 323	bug expand replaced with
<code>\@GLS@:</code> add <code>\glslabel,</code>	noexpand ..... 60
<code>\glsifplural, \glscapscase,</code>	<code>\@glsdisp:</code> add <code>\glslabel,</code>
<code>\glscustomtext</code> and	<code>\glsifplural, \glscapscase,</code>
<code>\glsinsert</code> ..... 104	<code>\glscustomtext</code> and
change to using <code>\glsentryfmt</code>	<code>\glsinsert</code> ..... 109
style commands ..... 104	change to using <code>\glsentryfmt</code>
removed <code>\MakeUppercase</code>	style commands ..... 109
(now moved to <code>\glsentryfmt</code> )	<code>\@glspl@:</code> add <code>\glslabel,</code>
..... 104	<code>\glsifplural, \glscapscase,</code>
<code>\@GLSpl:</code> add <code>\glslabel,</code>	<code>\glscustomtext</code> and
<code>\glsifplural, \glscapscase,</code>	<code>\glsinsert</code> ..... 105
<code>\glscustomtext</code> and	General: added <code>\glslabel,</code>
<code>\glsinsert</code> ..... 107	<code>\glsifplural, \glscapscase,</code>
change to using <code>\glsentryfmt</code>	<code>\glsinsert and \glscustomtext</code>
style commands ..... 108	..... 126–134
removed <code>\MakeUppercase</code>	changed to just use <code>\Glsentrydescplural</code>
as now dealt with in	..... 117
<code>\glsentryfmt</code> ..... 108	changed to just use <code>\glsentrydescplural</code>
<code>\@Gls@:</code> add <code>\glsifplural,</code>	..... 116, 117
<code>\glscapscase, \glscustomtext</code>	changed to just use <code>\Glsentrydesc</code>
and <code>\glsinsert</code> ..... 103	..... 115

changed to just use <code>\glentrydesc</code> ..... 115, 116	changed to just use <code>\glentryuservi</code> ..... 125
changed to just use <code>\Glsentryfirstplural</code> ..... 113	changed to just use <code>\Glsentryuserv</code> ..... 124
changed to just use <code>\glentryfirstplural</code> ..... 113, 114	changed to just use <code>\glentryuserv</code> ..... 124
changed to just use <code>\Glsentryfirst</code> ..... 111	Now requires textcase ..... 3
changed to just use <code>\glentryfirst</code> ..... 111, 112	acronymlists: replaced <code>\@addtoacronymlists</code> with <code>\DeclareAcronymList</code> .... 15
changed to just use <code>\Glsentryname</code> ..... 114	<code>\defglsgdisplay</code> : obsoleted .... 87
changed to just use <code>\glentryname</code> ..... 114, 115	<code>\defglsgdisplayfirst</code> : obso- leted ..... 88
changed to just use <code>\Glsentryplural</code> ..... 112	<code>\defglsgentryfmt</code> : new ..... 52
changed to just use <code>\glentryplural</code> ..... 112, 113	<code>\forglsgentries</code> : replaced <code>\ifx</code> with <code>\ifdefempty</code> ..... 47
changed to just use <code>\Glsentrysymbolplural</code> ..... 119	<code>\glsgassign@desc</code> : new ..... 67
changed to just use <code>\glentrysymbolplural</code> ..... 118, 119	<code>\glsgdefglossaryentry</code> : Fixed default counter if none sup- plied ..... 71
changed to just use <code>\Glsentrysymbol</code> ..... 118	<code>\glsgdoentryfmt</code> : new ..... 53
changed to just use <code>\glentrysymbol</code> ..... 117, 118	<code>\glsgdisplay</code> : obsoleted ..... 87
Changed to just use <code>\Glsentrytext</code> ..... 110	<code>\glsgdisplayfirst</code> : obsoleted .. 87
changed to just use <code>\glentrytext</code> ..... 110	<code>\glsgenentryfmt</code> : new ..... 82
changed to just use <code>\Glsentryuseriii</code> ..... 122	<code>\glsggetgrouptitle</code> : Added check in case non-Latin alpha- bet in use ..... 185
changed to just use <code>\glentryuseriii</code> ..... 122	<code>\glsglossarymark</code> : replaced <code>\MakeUppercase</code> with <code>\mfirstucMakeUppercase</code> . 35
changed to just use <code>\Glsentryuserii</code> ..... 121	<code>\glsgnavigation</code> : switched to us- ing <code>\@glsggetgrouptitle</code> 244
changed to just use <code>\glentryuserii</code> ..... 120, 121	<code>\ifglshasdesc</code> : replaced <code>\ifdefempty</code> with <code>\ifcsempy</code> ..... 50
changed to just use <code>\Glsentryuseriv</code> ..... 123	<code>\ifglshaslong</code> : new ..... 50
changed to just use <code>\glentryuseriv</code> ..... 123	<code>\ifglshasshort</code> : new ..... 51
changed to just use <code>\Glsentryuseri</code> ..... 120	<code>\ifglshassymbol</code> : replaced <code>\ifdefempty</code> with <code>\ifcsempy</code> ..... 50
changed to just use <code>\glentryuseri</code> ..... 119, 120	<code>\ifglsgused</code> : replaced <code>\ifthenelse</code> with <code>\ifbool</code> ..... 48
changed to just use <code>\Glsentryuservi</code> ..... 125	<code>\longnewglossaryentry</code> : new . 67
	<code>\newglossary</code> : replaced <code>\glsgdisplay</code> and <code>\glsgdisplayfirst</code> with <code>\glsgentryfmt</code> ..... 54
	compatible-3.07: cnew ..... 24
	<code>\SetCustomDisplayStyle</code> : up- dated to use <code>\defglsgentryfmt</code>

..... 225	..... 35
\SetDefaultAcronymDisplayStyle:	\glossarystyle: fixed bug
changed to use \defglentryfmt	caused by using \ifdef in-
..... 210	stead of \ifcsdef ..... 187
\SetDescriptionAcronymDisplayStyle:	\gls@assign@desc@field:
updated to use \defglentryfmt	changed to use \glsetnoexpandfield
..... 215	..... 17
\SetDescriptionDUAAcronymDisplayStyle:	\gls@assign@descplural@field:
updated to use \defglentryfmt	changed to use \glsetnoexpandfield
..... 214	..... 17
\SetDescriptionFootnoteAcronymDisplayStyle:	\gls@assign@name@field:
updated to use \defglentryfmt	changed to use \glsetnoexpandfield
..... 212	..... 17
\SetDUADisplayStyle: updated	\gls@assign@type@field:
to use \defglentryfmt .. 223	changed to use \glsetexpandfield
\SetFootnoteAcronymDisplayStyle:	..... 17
updated to use \defglentryfmt	\gls@checkseeallowed: new .. 57
..... 218	\glsaddallunused: set default to
\SetSmallAcronymDisplayStyle:	\@glo@types ..... 142
updated to use \defglentryfmt	\Glsentryfull: changed to use
..... 220	\acrfullformat ..... 139
\setupglossaries: new ..... 26	\glentryfull: changed to use
\showglolong: new ..... 231	\acrfullformat ..... 139
\showgloshort: new ..... 231	\Glsentryfullpl: changed to
numbers: new ..... 25	use \acrfullformat ..... 139
symbols: new ..... 25	\glentryfullpl: changed to
3.12a	use \acrfullformat ..... 139
\gls@defglossaryentry: added	\glsglossarymark: renamed
\glslabel ..... 68	\glossarymark to \glsglossarymark
\glsaddkey: new ..... 64	to avoid conflict with memoir 35
3.13a	\glsprestandardsort: new ... 10
\@gls@assign@symbol@field:	\glsetnoexpandfield: new .. 17
changed to use \glsetnoexpandfield	\altssuper4colheader: switched
..... 17	to \tabularnewline ..... 271
\@gls@assign@symbolplural@field:	\altssuper4colheaderborder:
changed to use \glsetnoexpandfield	switched to \tabularnewline
..... 17	..... 272
\@gls@link: removed \relax .. 91	long: switched to \tabularnewline
\@gls@notranslatorhook: new 21	..... 250, 251
\@gls@setupsort@standard:	long3col: switched to \tabularnewline
moved \@gls@santizesort	..... 252
to \glsprestandardsort .. 10	long3colheader: switched to
General: added cs@gls@notranslatorhook	\tabularnewline ..... 253
to else clause ..... 31	long3colheaderborder: switched
ucmark: added check for memoir . 9	to \tabularnewline ..... 253
see: added \gls@checkseeallowed	long4col: switched to \tabularnewline
..... 57	..... 253
\glossarysection: changed	long4colheader: switched to
\glossarymark to \glsglossarymark	\tabularnewline ..... 254



longheader: switched to \tabularnewline ..... 251	document class ..... 7
longheaderborder: switched to \tabularnewline ..... 251	\CustomAcronymFields: in- serted missing comma .... 226
\SetFootnoteAcronymDisplayStyle: fixed missing argument bug 218	4.02
super: switched to \tabularnewline ..... 266	\@acrfull: now using \acrfullfmt ..... 193
super3col: switched to \tabularnewline ..... 267	\@gls@indexdef: new ..... 26
super3colheader: switched to \tabularnewline ..... 268	\@gls@numbersdef: new ..... 25
super4col: switched to \tabularnewline ..... 269	\@gls@symbolsdef: new ..... 25
super4colheader: switched to \tabularnewline ..... 270	General: Removed \acronymfont ..... 130–134
super4colheaderborder: switched to \tabularnewline ..... 270	\ACRfullfmt: new ..... 194
superheader: switched to \tabularnewline ..... 267	\Acrfullfmt: new ..... 194
superheaderborder: switched to \tabularnewline ..... 267	\acrfullfmt: new ..... 193
3.14a	\ACRfullplfmt: new ..... 196
\@glswritefiles: renamed \glswritefiles to \@glswritefiles and used “savewrites” option to set \glswritefiles .... 156	\Acrfullplfmt: new ..... 195
General: new ..... 233	\acrfullplfmt: new ..... 195
acronyms: new ..... 13	\acronymentry: new ..... 198
\gls@defglossaryentry: added check for existence of default glossary ..... 69	sanitize: fixed bug that caused an error here ..... 21
set the default for firstplural to be the value of plural ..... 71	sc-short-long: new ..... 201
xindygloss: new ..... 24	sc-short-long-desc: new ... 203
\longprovideglossaryentry: new ..... 67	\Genacrfullformat: new ..... 86
compatible-2.07: added check for 2.07 before setting 3.07 compatibility ..... 25	\genacrfullformat: new ..... 86
notranslate: new ..... 21	\GenericAcronymFields: new 198
\provideglossaryentry: new . 62	\Genplacrfullformat: new ... 87
4.0	\genplacrfullformat: new ... 87
\gls@defglossaryentry: added check for first key ..... 71	\Glsentryfull: bug fix: added missing \acronymfont .... 139
4.01	\glentryfull: bug fix: added missing \acronymfont .... 139
General: fixed non-value options so that they can be passed to	\Glsentryfullpl: bug fix: added missing \acronymfont .... 139
	\glentryfullpl: bug fix: added missing \acronymfont .... 139
	\glsacrfmt: new ..... 84
	\GlsUseAcrEntryDisplayStyle: new ..... 199
	\GlsUseAcrStyleDefs: new .. 199
	short-long: new ..... 200
	short-long-desc: new ..... 202
	xindynoglsnumbers: new ..... 24
	sm-short-long: new ..... 201
	sm-short-long-desc: new ... 203
	\makeglossaries: made pream- ble only ..... 152
	index: new ..... 25
	\newacronymstyle: new ..... 198
	long-sc-short: new ..... 200

long-sc-short-desc: new ...	202	\@Gls@: removed \glslabel (defined in \@gls@link) ....	103
long-short: new .....	199	\@Gls@entry@field: new ....	134
long-short-desc: new .....	201	\@Glspl@: removed \glslabel (defined in \@gls@link) ...	106
long-sm-short: new .....	201	\@acrlong: removed \glslabel (defined in \@gls@link) ...	324
long-sm-short-desc: new ...	202	\@acrshort: removed \glslabel (defined in \@gls@link) ...	323
footnote: new .....	205	\@gls@: removed \glslabel (defined in \@gls@link) ....	101
footnote-desc: new .....	207	\@gls@access@display: new .	311
footnote-sc: new .....	207	\@gls@entry@field: new ....	134
footnote-sc-desc: new ....	207	\@gls@fetchfield: new .....	64
footnote-sm: new .....	207	\@gls@field@link: new ....	109
footnote-sm-desc: new ....	208	\@gls@link: added \glsdetoklabel	
\setacronymstyle: new ....	198	moved \@gls@link@opts and \@gls@link@label to \@gls@link .....	90
\SetDescriptionAcronymDisplayStyle: Moved check for empty custom text to prevent unwanted parenthetical material .....	216	\@gls@writedef: added \glsdetoklabel .....	63
\SetDescriptionFootnoteAcronymDisplayStyle: Moved check for empty custom text to prevent unwanted parenthetical material .....	212	\@glsdisp: removed \glslabel (defined in \@gls@link) ...	109
\SetFootnoteAcronymDisplayStyle: Moved check for empty custom text to prevent unwanted parenthetical material .....	218	\@glspl@: removed \glslabel (defined in \@gls@link) ...	105
\SetGenericNewAcronym: new	197	\@printglossary: added \glsdetoklabel .....	166
\SetSmallAcronymDisplayStyle: Moved check for empty custom text to prevent unwanted parenthetical material .....	221	General: changed default to \@empty instead of \relax ..	24
dua: new .....	203	removed \glslabel (defined in \@gls@link) .....	126
dua-desc: new .....	205	sc-short-long-desc: redefined to use accessibility information .....	329
numberedsection: added		\compatibleglossentry: added \glsdetoklabel .....	306
nameref option .....	6	\compatiblesubglossentry: added \glsdetoklabel ...	306
4.03		\Genacrfullformat: redefined to use accessibility information .....	322
\@@do@wrglossary: added		\genacrfullformat: redefined to use accessibility information .....	322
\glsdetoklabel .....	160	\Genplacrfullformat: redefined to use accessibility information .....	323
\@ACRlong: removed \glslabel (defined in \@gls@link) ...	325		
\@ACRshort: removed \glslabel (defined in \@gls@link) ...	324		
\@Acrlong: removed \glslabel (defined in \@gls@link) ...	324		
\@Acrshort: removed \glslabel (defined in \@gls@link) ...	323		
\@GLS@: removed \glslabel (defined in \@gls@link) ....	104		
\@GLSpl: removed \glslabel (defined in \@gls@link) ...	107		





\showglouserii: added \glsdetoklabel ..... 229	\@gls@getcounterprefix: added warning if no prefix can be formed ..... 161
\showglouseriii: added \glsdetoklabel ..... 229	\@gls@getothergrouptitle: new ..... 186
\showglouseriv: added \glsdetoklabel ..... 229	\@gls@noidx@do: new ..... 174
\showglouserv: added \glsdetoklabel ..... 229	\@gls@noref@warn: new ..... 156
\showglouservi: added \glsdetoklabel ..... 230	\@gls@reference: new ..... 176
dua: fixed bug in \acrfullfmt . 204	\@gls@warnonglossdefined: new ..... 16
fixed bug in \Acrfullplfmt . 205	\@gls@warnontheGLOSSdefined: new ..... 16
fixed bug in \acrfullplfmt . 205	\@no@makeGLOSSaries: new .. 156
redefined to use accessibility in- formation ..... 330	\@print@GLOSSary: new ..... 166
dua-desc: commented spurious EOL ..... 205	\@print@noidx@GLOSSary: new 173
redefined to use accessibility in- formation ..... 332	\@print@GLOSS@setsort: new . 165
4.04	\@print@GLOSSary: new ..... 165
\@gls@noidx@nosanitizesort: new ..... 18	General: added sort key to print- GLOSS group ..... 178
\@gls@noidx@sanitizesort: new ..... 18	\compatIBLEGLOSSentry: changed \newcommand to \def as is may or may not be defined ..... 306
\@gls@nosanitizesort: new . 18	\compatIBLESUBGLOSSentry: changed \newcommand to \def as is may or may not be defined ..... 306
\@gls@sanitizesort: new ... 18	\defGLSDisplayfirst: fixed un- wanted space ..... 88
\@glo@addchildren: new .... 168	\glo@grabfirst: new ..... 174
\@glo@do@sortentries: new . 169	\gls@defGLOSSaryentry: re- placed \ifx with \ifdefvoid 72
\@glo@grabfirst: new ..... 174	\glsnoidxdisplayloc: new .. 176
\@glo@sortedinsert: new ... 169	\glsnoidxdisplaylocLISThandler: new ..... 176
\@glo@sortentries: new .... 168	\glsnoidxlocLIST: new ..... 175
\@glo@sorthandler@case: new 170	\glsnoidxlocLISThandler: new ..... 176
\@glo@sorthandler@letter: new ..... 170	\glsnoidxstripaccents: new . 18
\@glo@sorthandler@nocase: new ..... 170	alttree: moved hangindent and parindent assignments out- side level test ..... 283
\@glo@sorthandler@word: new 170	\makeGLOSSaries: Moved def- inition of \glswrite to \makeGLOSSaries ..... 150
\@glo@sortmacro@case: new . 171	\makenoidxGLOSSaries: new . 152
\@glo@sortmacro@def: new .. 172	\printGLOSSary: changed to use new \@printGLOSSary ... 164
\@glo@sortmacro@def@do: new 172	\printnoidxGLOSSaries: new 165
\@glo@sortmacro@letter: new 171	
\@glo@sortmacro@nocase: new 172	
\@glo@sortmacro@standard: new ..... 171	
\@glo@sortmacro@use: new .. 173	
\@glo@sortmacro@word: new . 171	

<code>\printnoidxglossary:new</code> ..	164	.....	164
<code>\showgloclolist:new</code> .....	231		
<code>\warn@noprintglossary: Activate warning in \makeglossaries</code>		<code>\writeist: checked for definition of \glswrite</code> .....	144, 148

## Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.

Symbols	
\@do@wrglossary .....	160
\@do@wrglossary .....	159
\@glo@assign@sortkey .....	178
\@glossarysec .....	5
\@glossaryseclabel .....	6
\@glossarysecstar .....	6
\@gls@default@entryfmt ....	313
\@gls@expand@field .....	60
\@gls@fixbraces .....	162
\@gls@noidx@nosanitizesort .	18
\@gls@noidx@sanitizesort ...	18
\@gls@nosanitizesort .....	18
\@gls@sanitizesort .....	18
\@ACRlong .....	325
\@ACRshort .....	324
\@Acrlong .....	324
\@Acrshort .....	323
\@GLS@ .....	104
\@GLSpl .....	107
\@Gls@ .....	102
\@Gls@entry@field .....	134
\@Glspl@ .....	106
\@PGLS .....	239
\@PGLS@ .....	239
\@PGLSpl .....	239
\@PGLSpl@ .....	239
\@Pgls .....	237
\@Pgls@ .....	237
\@Pglspl .....	238
\@Pglspl@ .....	238
\@acrfull .....	193
\@acrlong .....	324
\@acrshort .....	323
\@addtoacronymlists .....	14
\@delimN .....	189
\@delimR .....	189
\@disable@onlypremakeg .....	28
\@disable@premakecs .....	28
\@disabled@gl saddx dy counters	39
\@do@seeglossary .....	161
\@do@wrglossary .....	158, 287
\@glo@addchildren .....	168
\@glo@default@sorttype .....	9
\@glo@do@sortentries .....	169
\@glo@grabfirst .....	174
\@glo@no@assign@sortkey ....	178
\@glo@seeautonumberlist .....	7
\@glo@sortedinsert .....	169
\@glo@sortentries .....	168
\@glo@sorthandler@case ....	170
\@glo@sorthandler@letter ...	170
\@glo@sorthandler@nocase ...	170
\@glo@sorthandler@word ....	170
\@glo@sortmacro@case .....	171
\@glo@sortmacro@def .....	172
\@glo@sortmacro@def@do ....	172
\@glo@sortmacro@letter ....	171
\@glo@sortmacro@nocase ....	172
\@glo@sortmacro@standard ...	171
\@glo@sortmacro@use .....	173
\@glo@sortmacro@word .....	171
\@glo@storeentry .....	74
\@glo@types .....	52
\@glossary .....	157
\@glossary@default@style ....	6
\@glossaryentryfield .....	73
\@glossarysection .....	36
\@glossarysubentryfield ....	73
\@gls .....	101
\@gls@ .....	101
\@gls@@link .....	89
\@gls@access@display .....	311
\@gls@addpredefinedattributes	
.....	40
\@gls@assign@symbol@field ...	17
\@gls@assign@symbolplural@field	
.....	17
\@gls@checkactual .....	98
\@gls@checkbar .....	97
\@gls@checkescactual .....	95
\@gls@checkescbar .....	96
\@gls@checkesclevel .....	96
\@gls@checkescquote .....	95
\@gls@checklevel .....	97
\@gls@checkmkidxchars .....	93

<code>\@gls@checkquote</code>	94	<code>\@gls@setupsort@use</code>	11
<code>\@gls@codepage</code>	46	<code>\@gls@startswithexpandonce</code>	61
<code>\@gls@counterwithin</code>	9	<code>\@gls@symbolsdef</code>	25
<code>\@gls@declareoption</code>	7	<code>\@gls@tmpb</code>	94
<code>\@gls@default@value</code>	56	<code>\@gls@toc</code>	37
<code>\@gls@do@acronymsdef</code>	13	<code>\@gls@updatechecked</code>	94
<code>\@gls@entry@field</code>	134	<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>	109	.....	42
<code>\@gls@fixbraces</code>	162	<code>\@gls@xdy@Lclass@Appendix-page-numbers</code>	
<code>\@gls@getcounter</code>	55	.....	42
<code>\@gls@getcounterprefix</code>	161	<code>\@gls@xdy@Lclass@Roman-page-numbers</code>	
<code>\@gls@getgrouptitle</code>	185	.....	42
<code>\@gls@getothergrouptitle</code>	186	<code>\@gls@xdy@Lclass@alpha-page-numbers</code>	
<code>\@gls@hypergroup</code>	243	.....	42
<code>\@gls@ifinlist</code>	38	<code>\@gls@xdy@Lclass@arabic-page-numbers</code>	
<code>\@gls@indexdef</code>	26	.....	42
<code>\@gls@keymap</code>	63, 234	<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>	41
<code>\@gls@loadtree</code>	8	<code>\@gls@xdycheckbackslash</code>	99
<code>\@gls@makefirsttuc</code>	241	<code>\@gls@xdycheckquote</code>	98
<code>\@gls@missingnumberlist</code>	59	<code>\@gls@alphacompositor</code>	32, 42
<code>\@gls@noaccess</code>	308	<code>\@gls@acronymlists</code>	14
<code>\@gls@noexpand@field</code>	60	<code>\@gls@defaultplural</code>	59
<code>\@gls@noexpand@fields</code>	60	<code>\@gls@defaultsort</code>	60
<code>\@gls@nohyperlist</code>	15	<code>\@gls@disp</code>	108
<code>\@gls@noidx@do</code>	174	<code>\@gls@firstletter</code>	143
<code>\@gls@noidx@setsanitizesort</code>	20	<code>\@gls@hypernumber</code>	189
<code>\@gls@noref@warn</code>	156	<code>\@gls@link</code>	100
<code>\@gls@notranslatorhook</code>	21	<code>\@gls@minrange</code>	143
<code>\@gls@numbersdef</code>	25	<code>\@gls@nextpages</code>	179
<code>\@gls@onlypremakeg</code>	28	<code>\@gls@nodesc</code>	59
<code>\@gls@provide@newglossary</code>	52	<code>\@gls@noname</code>	59
<code>\@gls@reference</code>	176	<code>\@gls@nonextpages</code>	178
<code>\@gls@renewglossary</code>	157	<code>\@gls@openfile</code>	150
<code>\@gls@sanitizedesc</code>	16	<code>\@gls@order</code>	23
<code>\@gls@sanitizename</code>	17	<code>\@gls@spl@</code>	105
<code>\@gls@sanitizesort</code>	18	<code>\@gls@target</code>	100
<code>\@gls@sanitizesymbol</code>	17	<code>\@gls@widestname</code>	283
<code>\@gls@saveentrycounter</code>	91	<code>\@gls@writefiles</code>	156
<code>\@gls@setacrstyle</code>	22	<code>\@istfilename</code>	32
<code>\@gls@setcounter</code>	54	<code>\@makeglossary</code>	149
<code>\@gls@setupsort@def</code>	11	<code>\@newglossary</code>	54
<code>\@gls@setupsort@standard</code>	10	<code>\@newglossaryentryposthook</code>	73



\@newglossaryentryprehook ...	73	\Acl .....	209
\@no@makeglossaries .....	156	\acl .....	208
\@no@post@desc .....	31	\Aclp .....	209
\@nopostdesc .....	31	\aclp .....	209
\@onlypremakeg .....	28	\Acp .....	209
\@p@glossarysection .....	36	\acp .....	209
\@pgls .....	236	\acrfootnote .....	211
\@pgls@ .....	236	\ACRfull .....	194
\@pglsp1 .....	236	\Acrfull .....	193
\@pglsp1@ .....	236	\acrfull .....	192, 193, 197, 206, 333
\@print@glossary .....	166	\ACRfullfmt .....	194
\@print@noidx@glossary .....	173	\Acrfullfmt .....	194
\@printgloss@setsort .....	165	\acrfullfmt .....	193
\@printglossary .....	165	\acrfullformat .....	84, 193
\@sPGLS .....	238	\ACRfullpl .....	195
\@sPGLSp1 .....	239	\Acrfullpl .....	195
\@sPgls .....	237	\acrfullpl .....	194
\@sPglspl .....	238	\ACRfullplfmt .....	196
\@set@glo@numformat .....	92, 288	\Acrfullplfmt .....	195
\@sgls .....	101, 108	\acrfullplfmt .....	195
\@sgls@link .....	89	\acrlinkfootnote .....	211
\@spgls .....	236	\acrlinkfullformat .....	193
\@spglsp1 .....	236	\ACRlong .....	131
\@wrglossary .....	158	\Acrlong .....	130
\@xdy@main@language .....	23	\acrlong .....	130
\@xdy@attributelist .....	38	\ACRlongpl .....	133
\@xdy@attributes .....	38	\Acrlongpl .....	132
\@xdy@language .....	45	\acrlongpl .....	132
\@xdy@lettergroups .....	46	\acrnameformat .....	196, 217
\@xdy@locationclassorder .....	44	\acrno1inkfootnote .....	211
\@xdy@locref .....	38	acronym (option) .....	13
\@xdy@requiredstyles .....	45	acronym styles:	
\@xdy@sortrules .....	44	dua .....	203, 330
\@xdy@useralphabets .....	41	dua-desc .....	205, 332
\@xdy@userlocationdefs .....	43	footnote .....	205, 332
\@xdy@userlocationnames .....	43	footnote-desc .....	207, 335
		footnote-sc .....	207, 334
		footnote-sc-desc .....	207, 335
		footnote-sm .....	207, 334
		footnote-sm-desc .....	208, 335
		long-sc-short .....	200
		long-sc-short-desc ..	202, 328
		long-short .....	199, 326
		long-short-desc .....	201, 328
		long-sm-short .....	201
		long-sm-short-desc ..	202, 329
		sc-short-long .....	201
		sc-short-long-desc ..	203, 329
		short-long .....	200, 327

## A

\Ac .....	209
\ac .....	209
access (key) .....	306
accsupp package .....	305
\accsuppglossaryentryfield .	326
\accsuppglossarysubentryfield	
.....	326
\Acf .....	209
\acf .....	209
\Acfp .....	209
\acfp .....	209

short-long-desc ..... 202, 329  
 sm-short-long ..... 201  
 sm-short-long-desc .. 203, 330  
 \acronymentry ..... 198  
 \acronymfont .....  
 ..... 84, 196, 213, 217, 220, 223  
 acronymlists (option) ..... 15  
 \acronymname ..... 28  
 acronyms (option) ..... 13  
 \acronymsort ..... 198  
 \acronymstype ..... 13, 191  
 \acrpluralsuffix ..... 192  
 \ACRshort ..... 127  
 \Acrshort ..... 126  
 \acrshort ..... 126  
 \ACRshortpl ..... 129  
 \Acrshortpl ..... 128  
 \acrshortpl ..... 128  
 \Acs ..... 208  
 \acs ..... 208  
 \Acsp ..... 208  
 \acsp ..... 208  
 \addglossarytocaptions ..... 29  
 \addto ..... 29  
 align (environment) ..... 75, 91  
 altlist (style) ..... 248  
 altlistgroup (style) ..... 248  
 altlisthypergroup (style) ..... 249  
 altlong4col (style) ..... 255  
 altlong4colborder (style) ..... 255  
 altlong4colheader (style) ..... 255  
 altlong4colheaderborder (style) 255  
 altlongragged4col (style) ..... 259  
 altlongragged4colborder (style) 260  
 altlongragged4colheader (style) 260  
 altlongragged4colheaderborder  
 (style) ..... 261  
 \altnewglossary ..... 54  
 altsuper4col (style) ..... 271  
 altsuper4colborder (style) .... 271  
 altsuper4colheader (style) .... 271  
 altsuper4colheaderborder  
 (style) ..... 271  
 altsuperragged4col (style) .... 276  
 altsuperragged4colborder  
 (style) ..... 277  
 altsuperragged4colheader  
 (style) ..... 277

altsuperragged4colheaderborder  
 (style) ..... 277  
 alttree (style) ..... 283  
 alttreegroup (style) ..... 285  
 alttreehypergroup (style) ..... 285  
 amsgen package ..... 4, 88  
 amsmath package ..... 75  
 \andname ..... 29  
 array package ..... 256, 272  
 article class ..... 161

## B

babel package ..... 27, 28, 30, 45, 342

## C

\capitalisewords ..... 241  
 \changes ..... 5  
 \compatglossarystyle ..... 292  
 compatible-2.07 (option) ..... 25  
 compatible-3.07 (option) ..... 24  
 \compatibleglossentry .. 182, 306  
 \compatiblesubglossentry 183, 306  
 counter (key) ..... 57  
 counter (option) ..... 15  
 \CustomAcronymFields ..... 226  
 \CustomNewAcronymDef ..... 226

## D

datatool package ..... 170  
 \DeclareAcronymList ..... 14  
 \DefaultNewAcronymDef .. 210, 336  
 \defentryfmt ..... 88  
 \defglstdisplay ..... 87  
 \defglstdisplayfirst ..... 88  
 \defglssentry ..... 54  
 \defglssentryfmt ..... 52, 55, 56, 79  
 \DefineAcronymSynonyms ..... 208  
 \delimN ..... 34, 188  
 \delimR ..... 34, 188  
 description (environment) .... 247  
 description (key) ..... 55  
 description (option) ..... 22  
 descriptionaccess (key) ..... 307  
 \DescriptionDUANewAcronymDef 214  
 \DescriptionFootnoteNewAcronymDef  
 ..... 212, 337  
 \descriptionname ..... 29  
 \DescriptionNewAcronymDef ..  
 ..... 216, 338  
 descriptionplural (key) ..... 55

descriptionpluralaccess (key) 307  
\detokenize ..... 48  
doc package ..... 4, 5, 12  
dua (acrstyle) ..... 203, 330  
dua (option) ..... 23  
dua-desc (acrstyle) ..... 205, 332  
\DUANewAcronymDef ..... 223

## E

entrycounter (option) ..... 9  
entrycounterwithin (option) .... 9  
\entryname ..... 29  
environments:  
    align ..... 75, 91  
    description ..... 247  
    longtable ..... 8, 227, 250–261  
    multicols ..... 262  
    supertabular ... 8, 227, 265–278  
    theglossary ..... 5, 16, 34,  
        181, 182, 187, 263, 264, 280–283  
    theindex ..... 278  
equation (counter) ..... 91, 92  
etoolbox package ..... 4, 240

## F

file types  
    .aux ..... 167  
    .glo ..... 74  
    .ist ..... 142, 143, 149  
    .toc ..... 37  
    .xdy ..... 32  
    glo ..... 233  
\findrootlanguage ..... 45  
first (key) ..... 56  
firstaccess (key) ..... 307  
\firstacronymfont ..... 84, 196  
firstplural (key) ..... 56  
firstpluralaccess (key) ..... 307  
footnote (acrstyle) ..... 205, 332  
footnote (option) ..... 22  
footnote-desc (acrstyle) .. 207, 335  
footnote-sc (acrstyle) .... 207, 334  
footnote-sc-desc (acrstyle) 207, 335  
footnote-sm (acrstyle) .... 207, 334  
footnote-sm-desc (acrstyle) 208, 335  
\FootnoteNewAcronymDef . 218, 339  
\forallglossaries ..... 47  
\forallglsentries ..... 47  
\forglsentries ..... 47

## G

garamondx package ..... 192  
\Genacrfullformat ..... 86, 322  
\genacrfullformat ..... 86, 322  
\GenericAcronymFields ..... 198  
\Genplacrfullformat ..... 87, 323  
\genplacrfullformat ..... 87, 323  
\glo@grabfirst ..... 174  
\glolinkprefix ..... 91  
glossareentry (counter) ..... 180  
glossaries package ..... 45,  
    46, 143, 227, 233, 247, 285, 305  
glossaries-accsupp package ... 73, 305  
\GlossariesWarning ..... 16  
\GlossariesWarningNoLine .... 16  
\glossary ..... 53, 149, 157, 186  
glossary counters:  
    glossaryentry ..... 179  
    glossarysubentry ..... 179  
glossary keys:  
    access ..... 306  
    counter ..... 57  
    description ..... 55  
    descriptionaccess ..... 307  
    descriptionplural ..... 55  
    descriptionpluralaccess . 307  
    first ..... 56  
    firstaccess ..... 307  
    firstplural ..... 56  
    firstpluralaccess ..... 307  
    long ..... 59  
    longaccess ..... 308  
    longplural ..... 59  
    longpluralaccess ..... 308  
    name ..... 55  
    nonumberlist ..... 58  
    parent ..... 57  
    plural ..... 56  
    pluralaccess ..... 307  
    see ..... 57  
    short ..... 58  
    shortaccess ..... 307  
    shortplural ..... 59  
    shortpluralaccess ..... 307  
    sort ..... 56  
    symbol ..... 56  
    symbolaccess ..... 307  
    symbolplural ..... 57  
    symbolpluralaccess ..... 307

text .....	56	altsuperragged4colborder	
textaccess .....	307	.....	277, 303
type .....	57	altsuperragged4colborder	277
user1 .....	58	altsuperragged4colheader	
user2 .....	58	.....	277, 303
user3 .....	58	altsuperragged4colheader	277
user4 .....	58	altsuperragged4colheaderborder	
user5 .....	58	.....	277, 303
user6 .....	58	altsuperragged4colheaderborder	
glossary package .....	1, 191	.....	277
glossary styles:		alttree ...	264, 282, 283, 285, 300
altlist .....	248, 249, 293	alttree .....	283
altlist .....	248	alttreegroup .....	285, 301
altlistgroup .....	248, 249, 294	alttreegroup .....	285
altlistgroup .....	248	alttreehypergroup ...	285, 301
altlisthypergroup ...	249, 294	alttreehypergroup .....	285
altlisthypergroup .....	249	index .....	262, 278–280, 297
altlong4col .....	255, 259, 296	index .....	278
altlong4col .....	255	indexgroup .....	279, 298
altlong4colborder ...	255, 296	indexgroup .....	279
altlong4colborder .....	255	indexhypergroup .....	279, 298
altlong4colheader ...	255, 296	indexhypergroup .....	279
altlong4colheader .....	255	inline .....	292
altlong4colheaderborder .		inline .....	244
.....	255, 296	list .....	6, 247–249, 293
altlong4colheaderborder .	255	list .....	247
altlongragged4col	259, 260, 297	listdotted .....	249, 294
altlongragged4col .....	259	listdotted .....	249
altlongragged4colborder .		listgroup .....	247, 248, 293
.....	260, 297	listgroup .....	247
altlongragged4colborder .	260	listhypergroup .....	248, 293
altlongragged4colheader .		listhypergroup .....	248
.....	260, 297	long .....	250–252, 256, 294, 296
altlongragged4colheader .	260	long .....	250
altlongragged4colheaderborder		long3col .....	252, 295
.....	261, 297	long3col .....	252
altlongragged4colheaderborder		long3colborder .....	252, 295
.....	261	long3colborder .....	252
altsuper4col .....	271, 276, 305	long3colheader .....	252, 295
altsuper4col .....	271	long3colheader .....	252
altsuper4colborder ..	271, 305	long3colheaderborder	253, 295
altsuper4colborder .....	271	long3colheaderborder ....	253
altsuper4colheader ..	271, 305	long4col .....	253–255, 295
altsuper4colheader .....	271	long4col .....	253
altsuper4colheaderborder		long4colborder .....	254, 295
.....	271, 305	long4colborder .....	254
altsuper4colheaderborder	271	long4colheader .....	254, 295
altsuperragged4col	276, 277, 303	long4colheader .....	254
altsuperragged4col .....	276	long4colheaderborder	254, 296

long4colheaderborder	....	254
longborder	.....	251, 294
longborder	.....	251
longheader	.....	251, 294
longheader	.....	251
longheaderborder	....	251, 295
longheaderborder	.....	251
longragged	.....	256–258
longragged	.....	256
longragged3col	...	258, 259, 297
longragged3col	.....	258
longragged3colborder		258, 297
longragged3colborder	....	258
longragged3colheader		259, 297
longragged3colheader	....	259
longragged3colheaderborder		
	.....	259, 297
longragged3colheaderborder		
	.....	259
longraggedborder	....	257, 296
longraggedborder	.....	257
longraggedheader	....	257, 296
longraggedheader	.....	257
longraggedheaderborder		257, 296
longraggedheaderborder	..	257
mcalttree	.....	264, 302
mcalttree	.....	264
mcalttreegroup	....	265, 302
mcalttreegroup	.....	264
mcalttreehypergroup		265, 302
mcalttreehypergroup	...	265
mcindex	.....	262, 301
mcindex	.....	262
mcindexgroup	.....	262, 301
mcindexgroup	.....	262
mcindexhypergroup	.	262, 301
mcindexhypergroup	....	262
mcmtree	.....	263, 301
mcmtree	.....	262
mcmtreegroup	.....	301
mcmtreegroup	.....	263
mcmtreehypergroup	..	263, 301
mcmtreehypergroup	.....	263
mcmtreeoname	.....	263, 301
mcmtreeoname	.....	263
mcmtreeonamegroup	.	264, 301
mcmtreeonamegroup	....	263
mcmtreeonamehypergroup		
	.....	264, 302
mcmtreeonamehypergroup		
	.....	264, 302
sublistdotted	.....	294
sublistdotted	.....	249
super	.....	265–267, 273, 303
super	.....	265
super3col	.....	267, 268, 304
super3col	.....	267
super3colborder	....	268, 304
super3colborder	.....	268
super3colheader	....	268, 304
super3colheader	.....	268
super3colheaderborder		268, 304
super3colheaderborder	...	268
super4col	.....	269–271, 304
super4col	.....	269
super4colborder	....	270, 305
super4colborder	.....	270
super4colheader	....	270, 305
super4colheader	.....	270
super4colheaderborder		270, 305
super4colheaderborder	...	270
superborder	.....	266, 304
superborder	.....	266
superheader	.....	266, 304
superheader	.....	266
superheaderborder	...	267, 304
superheaderborder	.....	267
superragged	.....	272, 274, 302
superragged	.....	272
superragged3col	..	274–276, 302
superragged3col	.....	274
superragged3colborder		275, 303
superragged3colborder	...	275
superragged3colheader		275, 303
superragged3colheader	...	275
superragged3colheaderborder		
	.....	276, 303
superraggedborder	...	273, 302
superraggedborder	.....	273
superraggedheader	...	273, 302
superraggedheader	.....	273
superraggedheaderborder	.	
	.....	274, 302
superraggedheaderborder	.	274
superraggedright3colheaderborder		
	.....	276
tree	.....	262, 280, 281, 283, 298
tree	.....	280
treegroup	.....	263, 281, 299

treegroup .....	280	\gls@assign@desc@field .....	17
treehypergroup .....	281, 299	\gls@assign@descplural@field .....	17
treehypergroup .....	281	\gls@assign@field .....	61
treenoname ...	263, 281, 282, 299	\gls@assign@name@field .....	17
treenoname .....	281	\gls@assign@type@field .....	17
treenonamegroup .....	282, 299	\gls@checkisacronymlist .....	15
treenonamegroup .....	282	\gls@checkseeallowed .....	57
treenonamehypergroup .....	282, 299	\gls@codepage .....	24
treenonamehypergroup .....	282	\gls@defglossaryentry .....	68
glossary-hypernav package .....	143	\gls@disablepagerefexpansion .....	158
glossary-list package .....	6, 8, 247	\gls@docclearpage .....	37
glossary-long package ..	8, 250, 259, 265	\gls@doentryfmt .....	53
glossary-longragged package .....	256	\gls@hypergroup prerun .....	243
glossary-mcols package .....	261	\gls@ifnotmeasuring .....	75
glossary-super package .....		\gls@level .....	60
.....	8, 250, 265, 272, 276	\gls@noidxglossary .....	156
glossary-superragged package .....	272	\gls@numberpage .....	159
glossary-tree package .....	8, 278	\gls@protected@pagefmts .....	158
glossaryentry (counter) ..	9, 180, 181	\gls@Romanpage .....	159
glossaryentry (counter) .....	179	\gls@romanpage .....	159
\glossaryentryfield ..	184, 187, 188	\gls@save@numberlist .....	163
\glossaryentrynumber ...	178, 179	\gls@suffixF .....	33
\glossaryentrynumbers .....		\gls@suffixFF .....	33
.....	7, 33, 165, 166	\glsaccessdisplay .....	313
\glossaryheader .....	182, 187	\glsaccsupp .....	311
\glossarymark .....	35	\glsadd .....	77, 141, 186
\glossaryname .....	28, 29	\glsadd options	
\glossarypostamble .....	35, 187	counter .....	141
\glossarypreamble .....	34, 187	format .....	141, 188
\glossarysection .....	6, 35, 53	\glsaddall .....	48, 77, 142
\glossarystyle .....	187, 227	\glsaddall options	
glossarysubentry (counter) ...		types .....	141, 142
.....	9, 180, 181	\glsaddallunused .....	142
glossarysubentry (counter) ...	179	\glsaddkey .....	64
\glossarysubentryfield .....	184	\GlsAddLetterGroup .....	46
\glossentry .....	56, 182	\GlsAddSortRule .....	44
\Glossentrydesc .....	183	\GlsAddXdyAlphabet .....	41
\glossentrydesc .....	183, 325	\GlsAddXdyAttribute .....	39, 286
\Glossentryname .....	182	\GlsAddXdyCounters .....	38, 286
\glossentryname .....	182, 325	\GlsAddXdyLocation .....	43, 287
\Glossentrysymbol .....	183	\GlsAddXdyStyle .....	45
\glossentrysymbol .....	183, 325	\glsautoprefix .....	6
\GLS .....	103	\glsclearpage .....	37
\Gls .....	102, 106, 240	\glsclsebrace .....	143
\gls .....	4, 56, 77, 89, 101,	\glscompositor .....	32, 43
103, 104, 109, 111–124, 180, 235		\glscounter .....	15, 54
\gls@Alphpage .....	159	\GlsDeclareNoHyperList .....	15
\gls@alphpage .....	159	\glsdefaulttype .....	12
\gls@assign@desc .....	67	\glsdefmain .....	12

<code>\GLSdesc</code> .....	116	<code>\glstentryname</code> .....	135, 163
<code>\Glsdesc</code> .....	115	<code>\glstentrynumberlist</code> ....	140, 154
<code>\glsdesc</code> .....	115, 116	<code>\Glstentryplural</code> .....	136
<code>\GLSdescplural</code> .....	117	<code>\glstentryplural</code> .....	135
<code>\Glsdescplural</code> .....	116	<code>\glstentrypluralaccess</code> .....	310
<code>\glsdescplural</code> .....	116, 117	<code>\Glstentryprefix</code> .....	235
<code>\glsdescriptionaccessdisplay</code>	312	<code>\glstentryprefix</code> .....	234
<code>\glsdescriptionpluralaccessdisplay</code>	312	<code>\Glstentryprefixfirst</code> .....	234
.....	312	<code>\glstentryprefixfirst</code> .....	234
<code>\glsdescwidth</code> ...	250, 256, 265, 272	<code>\Glstentryprefixfirstplural</code> .	235
<code>\glsdetoklabel</code> .....	48	<code>\glstentryprefixfirstplural</code> .	234
<code>\glsdisablehyper</code> .....	100	<code>\Glstentryprefixplural</code> .....	235
<code>\glsdisp</code> .....	108	<code>\glstentryprefixplural</code> .....	234
<code>\glsdisplay</code> .....	78, 87, 101	<code>\Glstentryshort</code> .....	138
<code>\glsdisplayfirst</code> .....	78, 87, 101	<code>\glstentryshort</code> .....	138
<code>\glsdisplaynumberlist</code>	140, 154, 176	<code>\glstentryshortaccess</code> .....	311
<code>\glsdoifexists</code> .....	48	<code>\Glstentryshortpl</code> .....	139
<code>\glsdoifexistsorwarn</code> .....	49	<code>\glstentryshortpl</code> .....	139
<code>\glsdoifnoexists</code> .....	49	<code>\glstentryshortpluralaccess</code> .	311
<code>\glsdoparenifnotempty</code> .....	220	<code>\glstentrysort</code> .....	137
<code>\glsenablehyper</code> .....	100	<code>\Glstentrysymbol</code> .....	136
<code>\glstentryaccess</code> .....	310	<code>\glstentrysymbol</code> .....	136
<code>\glstentrycounter</code> .....	91	<code>\glstentrysymbolaccess</code> .....	310
<code>\glstentrycounterlabel</code> .....	181	<code>\Glstentrysymbolplural</code> .....	136
<code>\Glstentrydesc</code> .....	135	<code>\glstentrysymbolplural</code> .....	136
<code>\glstentrydesc</code> .....	135	<code>\glstentrysymbolpluralaccess</code>	310
<code>\glstentrydescaccess</code> .....	310	<code>\Glstentrytext</code> .....	135
<code>\Glstentrydescplural</code> .....	135	<code>\glstentrytext</code> .....	48, 135, 163
<code>\glstentrydescplural</code> .....	135	<code>\glstentrytextaccess</code> .....	310
<code>\glstentrydescpluralaccess</code> ..	310	<code>\glstentrytype</code> .....	137
<code>\Glstentryfirst</code> .....	136	<code>\Glstentryuseri</code> .....	137
<code>\glstentryfirst</code> .....	136	<code>\glstentryuseri</code> .....	137
<code>\glstentryfirstaccess</code> .....	310	<code>\Glstentryuserii</code> .....	137
<code>\Glstentryfirstplural</code> .....	137	<code>\glstentryuserii</code> .....	137
<code>\glstentryfirstplural</code> .....	136	<code>\Glstentryuseriii</code> .....	138
<code>\glstentryfirstpluralaccess</code> .	310	<code>\glstentryuseriii</code> .....	137
<code>\glstentryfmt</code> .....	55, 56, 78	<code>\Glstentryuseriv</code> .....	138
<code>\Glstentryfull</code> .....	139	<code>\glstentryuseriv</code> .....	138
<code>\glstentryfull</code> ...	139, 197, 206, 334	<code>\Glstentryuserv</code> .....	138
<code>\Glstentryfullpl</code> .....	139	<code>\glstentryuserv</code> .....	138
<code>\glstentryfullpl</code> .....	139	<code>\Glstentryuservi</code> .....	138
<code>\glstentryitem</code> .....	181	<code>\glstentryuservi</code> .....	138
<code>\Glstentrylong</code> .....	139	<code>\glsexpandfields</code> .....	61
<code>\glstentrylong</code> .....	139	<code>\GLSfirst</code> .....	111
<code>\glstentrylongaccess</code> .....	311	<code>\Glsfirst</code> .....	111
<code>\Glstentrylongpl</code> .....	139	<code>\glsfirst</code> .....	111
<code>\glstentrylongpl</code> .....	139	<code>\glsfirstaccessdisplay</code> .....	312
<code>\glstentrylongpluralaccess</code> ..	311	<code>\GLSfirstplural</code> .....	113
<code>\Glstentryname</code> .....	135	<code>\Glsfirstplural</code> .....	113



<code>\glsfirstplural</code>	113	<code>\glsname</code>	114
<code>\glsfirstpluralaccessdisplay</code>	312	<code>\glsnameaccessdisplay</code>	312
<code>\glsgenacfmt</code>	84, 320	<code>\glsnamefont</code>	188
<code>\glsgenentryfmt</code>	82, 317	<code>\glsnavhyperlink</code>	242
<code>\glsgetgrouplabel</code>	186	<code>\glsnavhypertarget</code>	243
<code>\glsgetgrouptitle</code>	143, 185	<code>\glsnavigation</code>	244
<code>\glsglossarymark</code>	9, 35	<code>\glsnextpages</code>	179
<code>\glsgroupheading</code>	185, 187	<code>\glsnoexpandfields</code>	62
<code>\glsgroupskip</code>	185, 187, 247	<code>\glsnoidxdisplayloc</code>	176
<code>\glshyperlink</code>	141	<code>\glsnoidxdisplaylocclishandler</code>	176
<code>\glshypernavsep</code>	244	<code>\glsnoidxlocclishandler</code>	176
<code>\glshypernumber</code>	33, 188	<code>\glsnoidxlocclishandler</code>	176
<code>\glsIfListOfAcronyms</code>	14	<code>\glsnoidxnumberlistloophandler</code>	155
<code>\glsinlinedescformat</code>	246	<code>\glsnoidxstripaccents</code>	18
<code>\glsinlinedopostchild</code>	245, 246	<code>\glsnonextpages</code>	179
<code>\glsinlineemptydescformat</code>	246	<code>\glsnoxindywarning</code>	38
<code>\glsinlineformat</code>	246	<code>\glsnumberformat</code>	33
<code>\glsinlineparentchildseparator</code>	246	<code>\glsnumberlistloop</code>	155
<code>\glsinlinepostchild</code>	246	<code>\glsnumbersgroupname</code>	29, 143, 185
<code>\glsinlineseparator</code>	246	<code>\glsnumlistlastsep</code>	141
<code>\glsinlinesubdescformat</code>	246	<code>\glsnumlistsep</code>	141
<code>\glsinlinesubnameformat</code>	246	<code>\glsopenbrace</code>	143
<code>\glsinlinesubseparator</code>	246	<code>\glsorder</code>	23
<code>\glskeylisttok</code>	196	<code>\glsorg@endtheglossary</code>	5
<code>\glslabeltok</code>	196	<code>\glsorg@glossary</code>	4
<code>\glslink</code>	77, 78, 89, 101, 141, 186, 188	<code>\glsorg@theglossary</code>	5
<code>\glslink options</code>		<code>\glsorg@wrglossary</code>	4
<code>counter</code>	88, 101, 233	<code>\glspagelistwidth</code>	250, 256, 265, 272
<code>format</code>	89, 101, 188	<code>\glspar</code>	31
<code>hyper</code>	89, 101	<code>\GLSpl</code>	107
<code>local</code>	89	<code>\Glspl</code>	106, 240
<code>\glslistdottedwidth</code>	249	<code>\glspl</code>	77, 105–107
<code>\glslocalreset</code>	76	<code>\GLSplural</code>	112
<code>\glslocalresetall</code>	77	<code>\Glsplural</code>	112
<code>\glslocalunset</code>	76	<code>\glsplural</code>	112
<code>\glslocalunsetall</code>	77	<code>\glspluralaccessdisplay</code>	312
<code>\glslongaccessdisplay</code>	313	<code>\glspluralsuffix</code>	29, 56
<code>\glslongaccesskey</code>	341	<code>\glspostdescription</code>	8
<code>\glslongkey</code>	192	<code>\glspostinline</code>	246
<code>\glslongpluralaccessdisplay</code>	313	<code>\glsprestandardsort</code>	10
<code>\glslongpluralaccesskey</code>	341	<code>\glsquote</code>	143
<code>\glslongpluralkey</code>	192	<code>\glsrefentry</code>	180
<code>\glslongtok</code>	196	<code>\glsreset</code>	75
<code>\glsmakefirsttuc</code>	241	<code>\glsresetall</code>	76
<code>\glsmcols</code>	261	<code>\glsresetentrylist</code>	179
<code>\glsmoveentry</code>	73	<code>\glsresetsubentrycounter</code>	180
<code>\GLSname</code>	115	<code>\glssee</code>	162
<code>\Glsname</code>	114		





`\ifglshaschildren` ..... 49  
`\ifglshasdesc` ..... 50  
`\ifglshasfield` ..... 51  
`\ifglshaslong` ..... 50  
`\ifglshasparent` ..... 50  
`\ifglshasprefix` ..... 235  
`\ifglshasprefixfirst` ..... 235  
`\ifglshasprefixfirstplural` . 235  
`\ifglshasprefixplural` ..... 235  
`\ifglshasshort` ..... 51  
`\ifglshassymbol` ..... 50  
`\ifglstranslate` ..... 21  
`\ifglxsused` ..... 48, 75  
`\ifglxsindy` ..... 23  
`index (option)` ..... 25  
`index (style)` ..... 278  
`indexgroup (style)` ..... 279  
`indexhypergroup (style)` ..... 279  
`indexonlyfirst (option)` ..... 22  
`inline (style)` ..... 244  
`\inputencodingname` ..... 24  
`\istfile` ..... 156  
`\istfilename` ..... 32  
`\item` ..... 188, 247, 278, 279

## L

`link text` ..... 78  
`list (style)` ..... 247  
`listdotted (style)` ..... 249  
`listgroup (style)` ..... 247  
`listhypergroup (style)` ..... 248  
`\loadglsentries` ..... 12, 77  
`long (key)` ..... 59  
`long (style)` ..... 250  
`long-sc-short (acrstyle)` ..... 200  
`long-sc-short-desc (acrstyle)` ..  
..... 202, 328  
`long-short (acrstyle)` ..... 199, 326  
`long-short-desc (acrstyle)` . 201, 328  
`long-sm-short (acrstyle)` ..... 201  
`long-sm-short-desc (acrstyle)` ..  
..... 202, 329  
`long3col (style)` ..... 252  
`long3colborder (style)` ..... 252  
`long3colheader (style)` ..... 252  
`long3colheaderborder (style)` .. 253  
`long4col (style)` ..... 253  
`long4colborder (style)` ..... 254  
`long4colheader (style)` ..... 254

`long4colheaderborder (style)` .. 254  
`longaccess (key)` ..... 308  
`longborder (style)` ..... 251  
`longheader (style)` ..... 251  
`longheaderborder (style)` ..... 251  
`\longnewglossaryentry` .... 55, 67  
`longplural (key)` ..... 59  
`longpluralaccess (key)` ..... 308  
`\longprovideglossaryentry` ... 67  
`longragged (style)` ..... 256  
`longragged3col (style)` ..... 258  
`longragged3colborder (style)` .. 258  
`longragged3colheader (style)` .. 259  
`longragged3colheaderborder`  
`(style)` ..... 259  
`longraggedborder (style)` ..... 257  
`longraggedheader (style)` ..... 257  
`longraggedheaderborder (style)` 257  
`longtable (environment)` .....  
..... 8, 227, 250–261  
`longtable package` ..... 250, 256

## M

`\makefirsttuc` ..... 240  
`makeglossaries` 23, 32, 45, 46, 53, 167  
`\makeglossaries` .....  
..... 28, 32, 33, 52, 54, 150, 152  
`\makeglossary` ..... 152  
`makeindex` ..... 355  
`makeindex` ..... 10, 23, 29, 32–34,  
53, 54, 56, 74, 92, 95, 142, 145,  
148, 149, 160, 161, 184, 185, 287  
`delim_n` ..... 33  
`delim_r` ..... 34  
`page_compositor` ..... 32  
`special characters` ..... 93, 94, 142  
`makeindex (option)` ..... 23  
`\makenoidxglossaries` .... 20, 152  
`mcolalmtree (style)` ..... 264  
`mcolalmtreegroup (style)` ..... 264  
`mcolalmtreehypergroup (style)` . 265  
`mcolindex (style)` ..... 262  
`mcolindexgroup (style)` ..... 262  
`mcolindexhypergroup (style)` ... 262  
`mcoltree (style)` ..... 262  
`mcoltreegroup (style)` ..... 263  
`mcoltreehypergroup (style)` .... 263  
`mcoltreenoname (style)` ..... 263  
`mcoltreenonamegroup (style)` ... 263

mcoltreenonamehypergroup  
     (style) ..... 264  
 memoir class ..... 157  
 mfirstuc package ..... 1  
 \mfirstucMakeUppercase ..... 241  
 multicol package ..... 261  
 multicol (environment) ..... 262

## N

name (key) ..... 55  
 \new@glossaryentry ..... 62  
 \newacronym .....  
     ... 22, 23, 58, 59, 77, 78, 191, 192  
 \newacronymhook ..... 196  
 \newacronymstyle ..... 198  
 \newglossary 15, 53, 54, 149, 151, 177  
 \newglossaryentry .....  
     ..... 55, 62, 77, 78, 191, 192  
 \newglossaryentry options  
     access ..... 308, 310  
     counter ..... 57  
     description ..... 22, 55,  
         59, 62, 68, 115, 135, 192, 219, 307  
     descriptionaccess ..... 310, 312  
     descriptionplural ..... 116, 307  
     descriptionpluralaccess .. 310, 312  
     first ..... 56, 71,  
         101, 111, 136, 217, 222, 223, 307  
     firstaccess ..... 310, 312  
     firstplural ..... 56, 113, 136, 307  
     firstpluralaccess ..... 310, 312  
     format ..... 144  
     long ..... 84, 139, 308  
     longaccess ..... 311, 313  
     longplural ..... 139, 308  
     longpluralaccess ..... 311, 313  
     name ..... 55,  
         56, 59, 62, 68, 114, 134, 163, 306  
     nonumberlist ..... 58  
     parent ..... 57, 62  
     plural ..... 56, 71, 112, 307  
     pluralaccess ..... 310, 312  
     prefix ..... 234  
     prefixfirst ..... 234  
     prefixfirstplural ..... 234  
     prefixplural ..... 234  
     see ..... 7, 57, 151, 152  
     short ..... 84, 138, 307  
     shortaccess ..... 311, 313

shortplural ..... 139, 307  
 shortpluralaccess ..... 311, 313  
 sort ..... 56, 137, 184, 185  
 symbol ..... 55, 56, 117, 213,  
     215, 217, 222, 253, 269, 306–308  
 symbolaccess ..... 310, 312  
 symbolplural ..... 118, 307  
 symbolpluralaccess ..... 310, 312  
 text 56, 101, 109, 135, 213, 217, 307  
 textaccess ..... 310, 312  
 type ..... 12, 57, 77, 137  
 user1 ..... 119, 137, 308  
 user2 ..... 120, 137  
 user3 ..... 121, 137  
 user4 ..... 122, 138  
 user5 ..... 123, 138  
 user6 ..... 124, 138, 308

\newglossarystyle ..... 187  
 nogroupskip (option) ..... 9  
 nohypertypes (option) ..... 16  
 \noist ..... 149, 233, 292  
 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 ..... 191  
 order (option) ..... 23

## P

package options:  
     acronym ..... 12, 13, 28, 164, 192  
         true ..... 13  
     acronym ..... 13  
     acronymlists ..... 15  
     acronyms ..... 13  
     compatible-2.07 ..... 25

compatible-3.07 .....	24	savenumberlist .....	7
counter .....	15	savewrites .....	24, 360
counter .....	15	false .....	150
description .....	217, 218	true .....	150, 156
description .....	22	savewrites .....	24
dua .....	215, 217, 218	section .....	6, 36
dua .....	23	section .....	6
entrycounter .....	179	seeautonumberlist .....	7
true .....	9	shotcuts .....	23
entrycounter .....	9	smallcaps .....	22
entrycounterwithin .....	9	smaller .....	22
footnote .....	102–	sort	
106, 108, 109, 213, 215, 217, 219		def .....	10
footnote .....	22	standard .....	10
hyperfirst		use .....	10
false .....	102–106, 108, 109	sort .....	10
hyperfirst .....	22	style .....	7, 227
index .....	25	style .....	7
indexonlyfirst .....	362	subentrycounter .....	179
indexonlyfirst .....	22	subentrycounter .....	9
makeindex .....	146, 233	symbols .....	25
makeindex .....	23	toc .....	5
nogroupskip .....	9	true .....	5
nohypertypes .....	16	toc .....	5
nolist .....	227	translate .....	21
nolist .....	8	false .....	21
nolong .....	227, 250	translate .....	21
nolong .....	8	translator .....	21
nomain .....	12, 13	ucmark .....	9
nomain .....	13	xindy .....	23, 24, 146, 233
nonumberlist .....	7	xindy .....	24
nonumberlist .....	7	xindygloss .....	24
nopostdot .....	9	xindynoglsnumbers .....	24
noredefwarn .....	16	\pagelistname .....	29
nostyles .....	8	parent (key) .....	57
nosuper .....	227	\PGLS .....	238
nosuper .....	8	\Pgls .....	237
notranslate .....	21	\pgls .....	236
notree .....	227	\PGLSpl .....	239
notree .....	8	\Pglspl .....	238
nowarn .....	16	\pglspl .....	236
numberedsection .....	6	\phantomsection .....	35, 36
numberline .....	5	plural (key) .....	56
numberline .....	5	pluralaccess (key) .....	307
numbers .....	25	polyglossia package .....	27, 29
order .....	23	\printacronyms .....	13
sanitize .....	19, 55, 134, 135	\PrintChanges .....	5
sanitize .....	21	\printglossaries .....	
sanitizesort .....	20	.... 12, 34, 52, 55, 152, 164, 242	

<code>\printglossary</code> .....	34, 53, 152, 164, 166, 177, 242
<code>\printglossary options</code>	
<code>nogroupskip</code> .....	178
<code>nonumberlist</code> .....	178
<code>numberedsection</code> .....	177
<code>style</code> .....	177
<code>title</code> .....	177
<code>toctitle</code> .....	177
<code>type</code> .....	12, 163, 177
<code>\printnoidxglossaries</code> .....	165
<code>\printnoidxglossary</code> ....	164, 177
<code>\printnoidxglossary options</code>	
<code>sort</code> .....	178
<code>\provideglossaryentry</code> .....	62
<b>R</b>	
<code>\renewacronymstyle</code> .....	199
<code>\renewglossarystyle</code> .....	188
<code>\roman</code> .....	42
<b>S</b>	
<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> .....	201
<code>sc-short-long-desc (acrstyle)</code> ..	203, 329
<code>\scantokens</code> .....	48
<code>\section</code> .....	48
<code>section (option)</code> .....	6
<code>see (key)</code> .....	57
<code>seeautonumberlist (option)</code> ....	7
<code>\seename</code> .....	29
<code>\SetAcronymLists</code> .....	15
<code>\SetAcronymStyle</code> .....	14, 225
<code>\setacronymstyle</code> .....	198
<code>\SetCustomDisplayStyle</code> ....	225
<code>\SetCustomStyle</code> .....	226
<code>\SetDefaultAcronymDisplayStyle</code>	
.....	210
<code>\SetDefaultAcronymStyle</code> ....	211
<code>\SetDescriptionAcronymDisplayStyle</code>	
.....	215
<code>\SetDescriptionAcronymStyle</code> ..	217
<code>\SetDescriptionDUAAcronymDisplayStyle</code>	
.....	214
<code>\SetDescriptionDUAAcronymStyle</code>	
.....	215
<code>\SetDescriptionFootnoteAcronymDisplayStyle</code>	
.....	212
<code>\SetDescriptionFootnoteAcronymStyle</code>	
.....	213
<code>\SetDUADisplayStyle</code> .....	223
<code>\SetDUASStyle</code> .....	224
<code>\setentrycounter</code> .....	186
<code>\SetFootnoteAcronymDisplayStyle</code>	
.....	218
<code>\SetFootnoteAcronymStyle</code> ...	219
<code>\SetGenericNewAcronym</code> .....	197
<code>\setglossarypreamble</code> .....	34
<code>\setglossarysection</code> .....	36
<code>\setglossarystyle</code> .....	187
<code>\setglossentrycompatibility</code> ..	184
<code>\SetSmallAcronymDisplayStyle</code> ..	220
<code>\SetSmallAcronymStyle</code> .....	222
<code>\setStyleFile</code> .....	31, 32
<code>\setupglossaries</code> .....	26
<code>short (key)</code> .....	58
<code>short-long (acrstyle)</code> ....	200, 327
<code>short-long-desc (acrstyle)</code> .	202, 329
<code>shortaccess (key)</code> .....	307
<code>shortplural (key)</code> .....	59
<code>shortpluralaccess (key)</code> .....	307
<code>shotcuts (option)</code> .....	23
<code>\showacronymlists</code> .....	232
<code>\showglocounter</code> .....	229
<code>\showglodesc</code> .....	230
<code>\showglodescaccess</code> .....	342
<code>\showglodescplural</code> .....	230
<code>\showglodescpluralaccess</code> ...	342
<code>\showglofirst</code> .....	228
<code>\showglofirstaccess</code> .....	341
<code>\showglofirstpl</code> .....	228
<code>\showglofirstpluralaccess</code> ..	341
<code>\showgloflag</code> .....	231
<code>\showgloindex</code> .....	231
<code>\showglolevel</code> .....	228
<code>\showgloloclist</code> .....	231
<code>\showglolong</code> .....	231
<code>\showglolongaccess</code> .....	342
<code>\showglolongpluralaccess</code> ...	342
<code>\showgloname</code> .....	230
<code>\showglonameaccess</code> .....	341
<code>\showgloparent</code> .....	227
<code>\showgloplural</code> .....	228
<code>\showglopluralaccess</code> .....	341
<code>\showgloshort</code> .....	231



W	
\warn@nomakeglossaries .....	150
\warn@noprintglossary .....	164
\writeist 32, 39, 40, 43, 143, 286, 288	
X	
\xcapitalisewords .....	242
\xglaccsupp .....	311
xindy .....	355
xindy .....	10, 23, 24, 32, 38, 41, 43, 45, 46, 74, 98, 99, 143–145, 160, 167, 184, 233, 287
xindy (option) .....	24
xindygloss (option) .....	24
xindynoglsnumbers (option) .....	24
\xmakefirstuc .....	241
xspace package .....	4, 191