

# Documented Code For glossaries v3.04

Nicola L.C. Talbot

Dickimaw Books

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

2012-11-18

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 v3.04:  $\TeX$ 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.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</b>	<b>Main Package Code</b>	<b>3</b>
1.1	Package Definition	3
1.2	Package Options	5
1.3	Default values	19
1.4	Xindy	28
1.5	Loops and conditionals	38
1.6	Defining new glossaries	40
1.7	Defining new entries	43
1.8	Resetting and unsetting entry flags	55
1.9	Loading files containing glossary entries	56
1.10	Using glossary entries in the text	56
1.10.1	Links to glossary entries	58
1.10.2	Displaying entry details without adding information to the glossary	109
1.11	Adding an entry to the glossary without generating text	116
1.12	Creating associated files	116
1.13	Writing information to associated files	126
1.14	Glossary Entry Cross-References	130
1.15	Displaying the glossary	132
1.16	Acronyms	146
1.17	Predefined acronym styles	150
1.18	Predefined Glossary Styles	164
1.19	Debugging Commands	165
1.20	Compatibility with version 2.07 and below	169
<b>2</b>	<b>Mfirstuc Documented Code</b>	<b>170</b>
<b>3</b>	<b>Glossary Styles</b>	<b>172</b>
3.1	Glossary hyper-navigation definitions (glossary-hypernav package)	172
3.2	In-line Style (glossary-inline.sty)	174
3.3	List Style (glossary-list.sty)	176
3.4	Glossary Styles using longtable (the glossary-long package)	180
3.5	Glossary Styles using longtable (the glossary-longragged package)	185
3.6	Glossary Styles using multicols (glossary-mcols.sty)	191
3.7	Glossary Styles using supertabular environment (glossary-super package)	194
3.8	Glossary Styles using supertabular environment (glossary-superragged package)	201
3.9	Tree Styles (glossary-tree.sty)	206
<b>4</b>	<b>glossaries-compatible-207</b>	<b>214</b>

<b>5 Accessibility Support (glossaries-accsupp Code)</b>	<b>221</b>
5.1 Defining Replacement Text . . . . .	221
5.2 Accessing Replacement Text . . . . .	224
5.3 Displaying the Glossary . . . . .	237
5.4 Acronyms . . . . .	237
5.5 Debugging Commands . . . . .	241
<b>6 Multi-Lingual Support</b>	<b>242</b>
6.1 Babel Captions . . . . .	242
6.2 Polyglossia Captions . . . . .	248
6.3 Brazilian Dictionary . . . . .	252
6.4 Danish Dictionary . . . . .	252
6.5 Dutch Dictionary . . . . .	252
6.6 English Dictionary . . . . .	252
6.7 French Dictionary . . . . .	253
6.8 German Dictionary . . . . .	253
6.9 Irish Dictionary . . . . .	253
6.10 Italian Dictionary . . . . .	254
6.11 Magyar Dictionary . . . . .	254
6.12 Polish Dictionary . . . . .	254
6.13 Serbian Dictionary . . . . .	255
6.14 Spanish Dictionary . . . . .	255
<b>Glossary</b>	<b>255</b>
<b>Change History</b>	<b>255</b>
<b>Index</b>	<b>265</b>

## 1 Main Package Code

### 1.1 Package Definition

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

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries}[2012/11/18 v3.04 (NLCT)]
```

Required packages:

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

7 \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.)

```
8 \RequirePackage{amsgen}
```

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

```
9 \RequirePackage{etoolbox}
```

Check if doc has been loaded.

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

```
10 \newif\if@gls@docloaded
11 \@ifpackageloaded{doc}{%
12 {%
13   \@gls@docloadedtrue
14 }%
15 {%
16   \@ifclassloaded{nlctdoc}{\@gls@docloadedtrue}{\@gls@docloadedfalse}%
17 }
```

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

It 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
```

```
19 \newcommand{\glsorg@glossary}{%
20   \@bsphack
21   \begingroup
22     \@sanitize \glsorg@wrglossary
23 }
```

```
\glsorg@wrglossary
```

```
24 \newcommand{\glsorg@wrglossary}[1]{%
25   \protected@write\@glossaryfile{}{%
26     \string \glossaryentry{#1}{\thepage}}%
27   \endgroup
28   \@esphack
29 }
```

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

```
30 \let\glsorg@changes\changes
31 \renewcommand{\changes}[3]{%
32   \begingroup
33     \let\glossary\glsorg@glossary
34     \glsorg@changes{#1}{#2}{#3}%
35   \endgroup
36 }
```

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

`\glsorg@theglossary`

```
37 \let\glsorg@theglossary\theglossary
```

`sorg@endtheglossary`

```
38 \let\glsorg@endtheglossary\endtheglossary
```

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

```
39 \let\glsorg@PrintChanges\PrintChanges
40 \renewcommand{\PrintChanges}{%
41   \begingroup
42     \let\theglossary\glsorg@theglossary
43     \let\endtheglossary\glsorg@endtheglossary
44     \glsorg@PrintChanges
45   \endgroup
46 }
```

End of doc stuff.

```
47 \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.

```
48 \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`.

```
49 \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`.

```
50 \ifcsundef{chapter}%
51   {\newcommand*\@@glossarysec{section}}%
52   {\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 redefined `\glossarysection`.

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

Determine whether or not to use numbered sections.

```

\@glossarysecstar
56 \newcommand*{\@glossarysecstar}{*}

\@glossaryseclabel
57 \newcommand*{\@glossaryseclabel}{}

\glsautoprefix Prefix to add before label if automatically generated:
58 \newcommand*{\glsautoprefix}{}

numberedsection
59 \define@choicekey{glossaries.sty}{numberedsection}[\val\nr]{%
60 false,nolabel,autolabel}[nolabel]{%
61   \ifcase\nr\relax
62     \renewcommand*{\@glossarysecstar}{*}%
63     \renewcommand*{\@glossaryseclabel}{}%
64   \or
65     \renewcommand*{\@glossarysecstar}{}%
66     \renewcommand*{\@glossaryseclabel}{}%
67   \or
68     \renewcommand*{\@glossarysecstar}{}%
69     \renewcommand*{\@glossaryseclabel}{}%
70     \label{\glsautoprefix\glo@type}%
71   \fi
72 }

```

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](#).)

```

ssary@default@style
73 \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.
74 \define@key{glossaries.sty}{style}{%
75 \renewcommand*{\@glossary@default@style}{#1}}

```

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”:

```

lossaryentrynumbers
76 \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).

```

77 \DeclareOptionX{nonumberlist}{%
78   \renewcommand*{\glossaryentrynumbers}[1]{\gls@save@numberlist{#1}}%
79 }

```

`savenumberlist` Provide means to store the number list for entries.

```

80 \define@boolkey{glossaries.sty}{gls}{savenumberlist}[true]{}
81 \glssavenumberlistfalse

```

`o@seeautonumberlist`

```

82 \newcommand*{\glo@seeautonumberlist}{}

```

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

```

83 \DeclareOptionX{seeautonumberlist}{%
84   \renewcommand*{\glo@seeautonumberlist}{%
85     \def\glo@prefix{\glsnextpages}%
86   }%
87 }

```

`\@gls@loadlong`

```

88 \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 provided to reduce overhead caused by loading unrequired packages.

```

89 \DeclareOptionX{nolong}{\renewcommand*{\@gls@loadlong}{} }

```

`\@gls@loadsuper` The package isn't loaded if isn't installed.

```

90 \IfFileExists{supertabular.sty}{%
91   \newcommand*{\@gls@loadsuper}{\RequirePackage{glossary-super}}}%
92   \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 provided to reduce overhead caused by loading unrequired packages.

```

93 \DeclareOptionX{nosuper}{\renewcommand*{\@gls@loadsuper}{} }

```

`\@gls@loadlist`

```

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

```

`nolist` This option prevents from being loaded (to reduce overheads if required). Naturally, the styles defined in will not be available if this option is used.

```

95 \DeclareOptionX{nolist}{\renewcommand*{\@gls@loadlist}{} }

```

`\@gls@loadtree`

```

96 \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.
97 \DeclareOptionX{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).
98 \DeclareOptionX{nostyles}{%
99   \renewcommand*{\@gls@loadlong}{}%
100  \renewcommand*{\@gls@loadsuper}{}%
101  \renewcommand*{\@gls@loadlist}{}%
102  \renewcommand*{\@gls@loadtree}{}%
103  \let\@glossary@default@style\relax
104 }

\glspostdescription The description terminator is given by \glspostdescription (except for the
        3 and 4 column styles). This is a full stop by default:
105 \newcommand*{\glspostdescription}{\ifglsnopostdot\else.\fi}

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

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

ucmark Boolean option to determine whether or not to use \MakeUppercase in defini-
        tion of \glossarymark
110 \define@boolkey{glossaries.sty}[gls]{ucmark}[true]{}
111 \glsucmarkfalse

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.
112 \define@boolkey{glossaries.sty}[gls]{entrycounter}[true]{}
113 \glsentrycounterfalse

entrycounterwithin This option can be used to set a parent counter for glossaryentry. This option
        automatically sets entrycounter=true.
114 \define@key{glossaries.sty}{counterwithin}{%
115   \renewcommand*{\@gls@counterwithin}{#1}%
116   \glsentrycountertrue
117 }

```



`\@gls@counterwithin` The default value is no parent counter:

```
118 \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`.

```
119 \define@boolkey{glossaries.sty}[gls]{subentrycounter}[true]\{}
120 \glssubentrycounterfalse
```

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

```
121 \define@choicekey{glossaries.sty}{sort}{standard,def,use}\{}
122 \csname @gls@setupsort@#1\endcsname
123 }
```

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

```
124 \newcommand*\@gls@setupsort@standard\{}
    Store entry information when it's defined.
125 \def\do@glo@storeentry{\@glo@storeentry}%
    No count register required for standard sort.
126 \def\@gls@defsortcount##1\{}%
    Sort according to sort key (\@glo@sort) if provided otherwise sort according
    to the entry's name (\@glo@name).
127 \def\@gls@defsort##1##2\{}
128 \ifx\@glo@sort\@glsdefaultsort
129 \let\@glo@sort\@glo@name
130 \fi
131 \@gls@sanitizesort
132 \expandafter\protected@xdef\csname glo@##2@sort\endcsname{\@glo@sort}%
133 }\}
```

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

```
134 \def\@gls@setsort##1\{}%
135 }
```

Set standard sort as the default:

```
136 \@gls@setupsort@standard
```

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

```
137 \newcommand*\glssortnumberfmt[1]\{}
138 \ifnum#1<100000 0\fi
139 \ifnum#1<10000 0\fi
140 \ifnum#1<1000 0\fi
141 \ifnum#1<100 0\fi
142 \ifnum#1<10 0\fi
143 \number#1%
144 }
```

```

\@gls@setupsort@def  Set up the macros for order of definition sorting.
145 \newcommand*{\@gls@setupsort@def}{%
    Store entry information when it's defined.
146   \def\do@glo@storeentry{\@glo@storeentry}%
    Defined count register associated with the glossary.
147   \def\@gls@defsortcount##1{%
148     \expandafter\global
149     \expandafter\newcount\csname glossary@##1@sortcount\endcsname
150   }%
    Increment count register associated with the glossary and use as the sort key.
151   \def\@gls@defsort##1##2{%
152     \expandafter\global\expandafter
153     \advance\csname glossary@##1@sortcount\endcsname by 1\relax
154     \expandafter\protected\edef\csname glo@##2@sort\endcsname{%
155       \expandafter\glssortnumberfmt
156       {\csname glossary@##1@sortcount\endcsname}}%
157   }%
    Don't need to do anything when the entry is used.
158   \def\@gls@setsort##1{%
159 }

\@gls@setupsort@use  Set up the macros for order of use sorting.
160 \newcommand*{\@gls@setupsort@use}{%
    Don't store entry information when it's defined.
161   \let\do@glo@storeentry\@gobble
    Defined count register associated with the glossary.
162   \def\@gls@defsortcount##1{%
163     \expandafter\global
164     \expandafter\newcount\csname glossary@##1@sortcount\endcsname
165   }%
    Initialise the sort key to empty.
166   \def\@gls@defsort##1##2{%
167     \expandafter\gdef\csname glo@##2@sort\endcsname{%
168   }%
    If the sort key hasn't been set, increment the counter associated with the glossary and set the sort key.
169   \def\@gls@setsort##1{%
    Get the parent, if one exists
170     \edef\@glo@parent{\csname glo@##1@parent\endcsname}%
    Set the information for the parent entry if not already done.
171     \ifx\@glo@parent\@empty
172     \else
173       \expandafter\@gls@setsort\expandafter{\@glo@parent}%
174     \fi

```

Set index information for this entry

```

175 \edef\@glo@type{\csname glo###1@type\endcsname}%
176 \edef\@gls@tmp{\csname glo###1@sort\endcsname}%
177 \ifx\@gls@tmp\@empty
178   \expandafter\global\expandafter
179   \advance\csname glossary@\@glo@type @sortcount\endcsname by 1\relax
180   \expandafter\protected@xdef\csname glo###1@sort\endcsname{%
181     \expandafter\glssortnumberfmt
182     {\csname glossary@\@glo@type @sortcount\endcsname}}%
183   \@glo@storeentry{##1}%
184 \fi
185 }%
186 }

```

`\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.)

```

187 \newcommand*{\glsdefmain}{%
188   \if@gls@docloaded
189     \newglossary[glg2]{main}{gls2}{glo2}{\glossaryname}%
190   \else
191     \newglossary{main}{gls}{glo}{\glossaryname}%
192   \fi
193 }

```

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`

```

194 \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`

```

195 \newcommand*{\acronymtype}{\glsdefaulttype}

```

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

```

196 \DeclareOptionX{nomain}{%
197   \let\glsdefaulttype\relax
198   \renewcommand*{\glsdefmain}{}%
199 }

```

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.

```
200 \define@boolkey{glossaries.sty}[gls]{acronym}[true]{%
201   \DeclareAcronymList{acronym}%
202 }
```

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

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

\@addtoacronymlists

```
204 \newcommand*{\@addtoacronymlists}[1]{%
205   \ifx\@glsacronymlists\@empty
206     \protected@xdef\@glsacronymlists{#1}%
207   \else
208     \protected@xdef\@glsacronymlists{\@glsacronymlists,#1}%
209   \fi
210 }
```

\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.)

```
211 \newcommand*{\DeclareAcronymList}[1]{%
212   \glsIfListOfAcronyms{#1}{\@addtoacronymlists{#1}}%
213 }
```

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

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

```
214 \newcommand{\glsIfListOfAcronyms}[1]{%
215   \edef\@do@glis@islistofacronyms{%
216     \noexpand\@glis@islistofacronyms{#1}{\@glsacronymlists}}%
217   \@do@glis@islistofacronyms
218 }
```

Internal command requires label and list to be expanded:

```
219 \newcommand{\@glis@islistofacronyms}[4]{%
220   \def\@glis@islistofacronyms##1,#1,##2\end@glis@islistofacronyms{%
221     \def\@before{##1}\def\@after{##2}}%
222   \glis@islistofacronyms,#2,#1,\@nil\end@glis@islistofacronyms
223   \ifx\@after\@nnil
```

Not found

```
224   #4%
225   \else
```

Found

```
226    #3%
227    \fi
228 }
```

`\if@glsisacronymlist` Convenient boolean.

```
229 \newif\if@glsisacronymlist
```

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

```
230 \newcommand*{\glsc@checkisacronymlist}[1]{%
231     \glscIfListOfAcronyms{#1}%
232     {\@glsisacronymlisttrue}{\@glsisacronymlistfalse}%
233 }
```

`\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.)

```
234 \newcommand*{\SetAcronymLists}[1]{%
235     \renewcommand*{\@glsacronymlists}{#1}%
236 }
```

`acronymlists`

```
237 \define@key{glossaries.sty}{acronymlists}{%
238     \@addtoacronymlists{#1}%
239 }
```

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`

```
240 \newcommand{\glscounter}{page}
```

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

```
241 \define@key{glossaries.sty}{counter}{%
242     \renewcommand*{\glscounter}{#1}%
243 }
```

The glossary keys whose values are written to another file (i.e. sort, name, description and symbol) need to be sanitized, otherwise fragile commands would not be able to be used in `\newglossaryentry`. However, strange results will occur if you then use those fields in the document. As these fields are not normally used in the document, but are by default only used in the glossary, the default is to sanitize them. If however you want to use these values in the document (either by redefining commands like `\glsdisplay` or by using commands like `\glsentrydesc`) you will have to switch off the sanitization using

the sanitize package option, but you will then have to use `\protect` to protect fragile commands when defining new glossary entries. The `sanitize` option takes a key-value list as its value, which can be used to switch individual values on and off. For example:

```
\usepackage[sanitize={description,name,symbol=false}]{glossaries}
```

will switch off the sanitization for the `symbol` key, but switch it on for the `description` and `name` keys. This would mean that you can use fragile commands in the `description` and `name` when defining a new glossary entry, but not for the `symbol`.

The default values are defined as:

```
\@gls@sanitizedesc
244 \newcommand*{\@gls@sanitizedesc}{\@onelevel@sanitize\@glo@desc}

\@gls@sanitizename
245 \newcommand*{\@gls@sanitizename}{\@onelevel@sanitize\@glo@name}

\@gls@sanitizesymbol
246 \newcommand*{\@gls@sanitizesymbol}{\@onelevel@sanitize\@glo@symbol}

\@gls@sanitizesort
247 \newcommand*{\@gls@sanitizesort}{\@onelevel@sanitize\@glo@sort}
```

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.

Firstly the `description`. If set, it will redefine `\@gls@sanitizedesc` to use `\@onelevel@sanitize`, otherwise `\@gls@sanitizedesc` will do nothing.

```
248 \define@boolkey[gl]{sanitize}{description}[true]{%
249 \ifgl@sanitize@description
250   \renewcommand*{\@gls@sanitizedesc}{\@onelevel@sanitize\@glo@desc}%
251 \else
252   \renewcommand*{\@gls@sanitizedesc}{}%
253 \fi
254 }
```

Similarly for the `name` key:

```
255 \define@boolkey[gl]{sanitize}{name}[true]{%
256 \ifgl@sanitize@name
257   \renewcommand*{\@gls@sanitizename}{\@onelevel@sanitize\@glo@name}%
258 \else
259   \renewcommand*{\@gls@sanitizename}{}%
260 \fi}
```

and for the `symbol` key:

```
261 \define@boolkey[gl]{sanitize}{symbol}[true]{%
262 \ifgl@sanitize@symbol
```

```

263 \renewcommand*{\@gls@sanitizesymbol}{%
264 \@onelevel@sanitize\@glo@symbol}%
265 \else
266 \renewcommand*{\@gls@sanitizesymbol}{}%
267 \fi}

and for the sort key:
268 \define@boolkey[gls]{sanitize}{sort}[true]{%
269 \ifgls@sanitize@sort
270 \renewcommand*{\@gls@sanitizesort}{%
271 \@onelevel@sanitize\@glo@sort}%
272 \else
273 \renewcommand*{\@gls@sanitizesort}{}%
274 \fi}

```

**sanitize** Now define the sanitize option. It can either take a key-val list as its value, or it can take the keyword none, which is equivalent to description=false, symbol=false, name=false:

```

275 \define@key{glossaries.sty}{sanitize}[description=true,symbol=true,
276 name=true]{%
277 \ifthenelse{\equal{#1}{none}}{%
278 {%
279 \renewcommand*{\@gls@sanitizedesc}{}%
280 \renewcommand*{\@gls@sanitizename}{}%
281 \renewcommand*{\@gls@sanitizesymbol}{}%
282 }%
283 {%
284 \setkeys[gls]{sanitize}{#1}}%
285 }

```

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

```

286 \define@boolkey{glossaries.sty}[gls]{translate}[true]{%

```

Set the default value:

```

287 \glstranslatefalse
288 \ifpackageloaded{translator}%
289 {\glstranslatetrue}%
290 {%
291 \ifpackageloaded{polyglossia}%
292 {\glstranslatetrue}%
293 {%
294 \ifpackageloaded{babel}{\glstranslatetrue}{}}%
295 }%
296 }

```

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

```

297 \define@boolkey{glossaries.sty}[gls]{indexonlyfirst}[true]{%
298 \glsindexonlyfirstfalse

```

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

```

299 \define@boolkey{glossaries.sty}[gls]{hyperfirst}[true]{}
300 \glshyperfirsttrue

```

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

```

301 \define@boolkey{glossaries.sty}[glsacr]{footnote}[true]{%
302 \ifthenelse{\boolean{glsacrdescription}}{}{}%
303 {\renewcommand*{\@gls@sanitizedesc}{}}%
304 }

```

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

```

305 \define@boolkey{glossaries.sty}[glsacr]{description}[true]{%
306 \renewcommand*{\@gls@sanitizesymbol}{}}%
307 }

```

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

```

308 \define@boolkey{glossaries.sty}[glsacr]{smallcaps}[true]{%
309 \renewcommand*{\@gls@sanitizesymbol}{}}%
310 }

```

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

```

311 \define@boolkey{glossaries.sty}[glsacr]{smaller}[true]{%
312 \renewcommand*{\@gls@sanitizesymbol}{}}%
313 }

```

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

```

314 \define@boolkey{glossaries.sty}[glsacr]{dua}[true]{%
315 \renewcommand*{\@gls@sanitizesymbol}{}}%
316 }

```

**shortcuts** Define acronym shortcuts.

```

317 \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.

```

318 \newcommand*{\glsorder}{word}

```

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

```

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

```

**order**

```

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

```



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

```
322 \newif\ifglxindy
```

The default is `makeindex`:

```
323 \glxindyfalse
```

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

```
324 \DeclareOptionX{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 `glxnumbers` determines whether to automatically add the `glxnumbers` letter group.

```
325 \define@boolkey[glx]{xindy}{glxnumbers}[true]{}
326 \glx@xindy@glxnumberstrue
```

`\@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.)

```
327 \def\@xdy@main@language{\rootlanguagename}%
```

Define key to set the language

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

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

```
329 \ifcsundef{inputencodingname}{%
330   \def\glx@codepage{}}{%
331   \def\glx@codepage{\inputencodingname}
332 }
```

Define a key to set the code page.

```
333 \define@key[glx]{xindy}{codepage}{\def\glx@codepage{#1}}
```

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

```
334 \define@key{glossaries.sty}{xindy}[]{%
335   \glxindytrue
336   \setkeys[glx]{xindy}{#1}%
337 }
```

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

```
338 \define@boolkey{glossaries.sty}[glx]{savewrites}[true]{}

Set default:
339 \glssavewritesfalse
```

```

\GlossariesWarning Prints a warning message.
340 \newcommand*\GlossariesWarning}[1]{%
341   \PackageWarning{glossaries}{#1}%
342 }

sariesWarningNoLine Prints a warning message without the line number.
343 \newcommand*\GlossariesWarningNoLine}[1]{%
344   \PackageWarningNoLine{glossaries}{#1}%
345 }

Define package option to suppress warnings
346 \DeclareOptionX{nowarn}{%
347   \renewcommand*\GlossariesWarning}[1]{}%
348   \renewcommand*\GlossariesWarningNoLine}[1]{}%
349 }

compatible-2.07
350 \define@boolkey{glossaries.sty}[gls]{compatible-2.07}[true]{}
351 \csname glscompatible-2.07false\endcsname

Process package options:
352 \ProcessOptionsX

If package is loaded, check to see if is installed, but only if translation is re-
quired.
353 \ifglstranslate
354   \ifpackageloaded{polyglossia}%
355   {%
polyglossia fakes babel so need to check for polyglossia first.
356   }%
357   {%
358     \ifpackageloaded{babel}%
359     {%
360       \IfFileExists{translator.sty}%
361       {%
362         \RequirePackage{translator}%
363       }%
364     }%
365   }%
366   {}
367 }
368 \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}`).

```

369 \ifthenelse{\equal{\glscounter}{section}}{%
370 {%
371   \ifcsundef{chapter}{}%
372   {%
373     \let\@gls@old@chapter\@chapter
374     \def\@chapter[#1]#2{\@gls@old@chapter[#1]{#2}%
375     \ifcsundef{hyperdef}{}{\hyperdef{section}{\thesection}{}}}%
376   }%
377 }%
378 {}

```

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

```

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

```

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

```

380 \newcommand*{\@onlypremakeg}[1]{%
381 \ifx\@gls@onlypremakeg\@empty
382   \def\@gls@onlypremakeg{#1}%
383 \else
384   \expandafter\toks@\expandafter{\@gls@onlypremakeg}%
385   \edef\@gls@onlypremakeg{\the\toks@,\noexpand#1}%
386 \fi}

```

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

```

387 \newcommand*{\@disable@onlypremakeg}{%
388 \@for\@thiscs:=\@gls@onlypremakeg\do{%
389   \expandafter\@disable@premakecs\@thiscs%
390 }}

```

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

```

391 \newcommand*{\@disable@premakecs}[1]{%
392   \def#1{\PackageError{glossaries}{\string#1\space may only be
393   used before \string\makeglossaries}{You can't use
394   \string#1\space after \string\makeglossaries}}%
395 }

```

### 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`

396 `\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`

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

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

398 `\newcommand*{\glstocctitle}[1]{%`  
399 `\def\glossarytocctitle{\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`

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

`\descriptionname`

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

`\symbolname`

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

`\pagelistname`

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

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

`glssymbolsgroupname`

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

`glsnumbersgroupname`

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

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

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

`\seename`

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

`\andname`

408 `\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 ba-
                      bel is also loaded, it will redefine \glossaryname whenever the language is set,
                      so override it. (Don't use \addto as doesn't define it.)
409 \newcommand*{\addglossarytocaptions}[1]{%
410   \ifcsundef{captions#1}{}%
411   {%
412     \expandafter\let\expandafter\@gls@tmp\csname captions#1\endcsname
413     \expandafter\toks@\expandafter{\@gls@tmp
414     \renewcommand*{\glossaryname}{\translate{Glossary}}}%
415   }%
416   \expandafter\edef\csname captions#1\endcsname{\the\toks@}%
417 }%
418 }

419 \ifglstranslate

  If is not install, used standard captions, otherwise load dictionary.
420 \@ifpackageloaded{translator}{%
421   \usedictionary{glossaries-dictionary}%
422   \addglossarytocaptions{portuges}%
423   \addglossarytocaptions{portuguese}%
424   \addglossarytocaptions{brazil}%
425   \addglossarytocaptions{brazilian}%
426   \addglossarytocaptions{danish}%
427   \addglossarytocaptions{dutch}%
428   \addglossarytocaptions{afrikaans}%
429   \addglossarytocaptions{english}%
430   \addglossarytocaptions{UKenglish}%
431   \addglossarytocaptions{USenglish}%
432   \addglossarytocaptions{american}%
433   \addglossarytocaptions{australian}%
434   \addglossarytocaptions{british}%
435   \addglossarytocaptions{canadian}%
436   \addglossarytocaptions{newzealand}%
437   \addglossarytocaptions{french}%
438   \addglossarytocaptions{frenchb}%
439   \addglossarytocaptions{français}%
440   \addglossarytocaptions{acadian}%
441   \addglossarytocaptions{canadien}%
442   \addglossarytocaptions{german}%
443   \addglossarytocaptions{germanb}%
444   \addglossarytocaptions{austrian}%
445   \addglossarytocaptions{naustrian}%
446   \addglossarytocaptions{ngerman}%
447   \addglossarytocaptions{irish}%
448   \addglossarytocaptions{italian}%
449   \addglossarytocaptions{magyar}%

```

```

450 \addglossarytocaptions{hungarian}%
451 \addglossarytocaptions{polish}%
452 \addglossarytocaptions{spanish}%
453 \renewcommand*{\glsettoctitle}[1]{%
454 \ifthenelse{\equal{#1}{main}}{%
455 \translatelet{\glossarytoctitle}{Glossary}}{%
456 \ifthenelse{\equal{#1}{acronym}}{%
457 \translatelet{\glossarytoctitle}{Acronyms}}{%
458 \def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}}%
459 \renewcommand*{\glossaryname}{\translate{Glossary}}%
460 \renewcommand*{\acronymname}{\translate{Acronyms}}%
461 \renewcommand*{\entryname}{\translate{Notation (glossaries)}}%
462 \renewcommand*{\descriptionname}{%
463 \translate{Description (glossaries)}}%
464 \renewcommand*{\symbolname}{\translate{Symbol (glossaries)}}%
465 \renewcommand*{\pagelistname}{%
466 \translate{Page List (glossaries)}}%
467 \renewcommand*{\glssymbolsgroupname}{%
468 \translate{Symbols (glossaries)}}%
469 \renewcommand*{\glsnumbersgroupname}{%
470 \translate{Numbers (glossaries)}}%
471 }{%

472 \@ifpackageloaded{polyglossia}%
473 {\RequirePackage{glossaries-polyglossia}}%
474 {%
475 \@ifpackageloaded{babel}{%
476 \RequirePackage{glossaries-babel}}}%
477 }}
478 \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.

```
479 \newcommand*\nopostdesc{}
```

`\@nopostdesc` Suppress next description terminator.

```

480 \newcommand*\@nopostdesc{%
481 \let\org@glspostdescription\glspostdescription
482 \def\glspostdescription{%
483 \let\glspostdescription\org@glspostdescription}%
484 }

```

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

```
485 \newcommand{\glspar}{\par}
```

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

```

486 \ifglxindy
487 \newcommand{\setStyleFile}[1]{%
488 \renewcommand{\istfilename}{#1.xdy}}

```

```

489 \else
490   \newcommand{\setStyleFile}[1]{%
491     \renewcommand{\istfilename}{#1.ist}}
492 \fi

```

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

```

493 \@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`

```

494 \ifglxindy
495   \def\istfilename{\jobname.xdy}
496 \else
497   \def\istfilename{\jobname.ist}
498 \fi

```

The `makeglossaries` Perl script picks up this name from the auxiliary file. If the name ends with `.xdy` it calls `xindy` otherwise it calls `makeindex`. Since its not required by  $\TeX$ , `\@istfilename` ignores its argument.

`\@istfilename`

```

499 \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`

```

500 \newcommand*\glscompositor{.}

```

`\glsSetCompositor` Sets the compositor.

```

501 \newcommand*\glsSetCompositor}[1]{%
502   \renewcommand*\glscompositor{#1}}

```

Only use before `\makeglossaries`

```

503 \@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.

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

`\glsSetAlphaCompositor` Sets the alpha compositor.

```
505 \ifglsexindy
506   \newcommand*\glsSetAlphaCompositor[1]{%
507     \renewcommand*\@glsAlphacompositor{#1}}
508 \else
509   \newcommand*\glsSetAlphaCompositor[1]{%
510     \glsnoxindywarning\glsSetAlphaCompositor}
511 \fi
```

Can only be used before `\makeglossaries`

```
512 \@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.

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

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

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

Only has an effect when used before `\makeglossaries`

```
516 \@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.

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

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

```
518 \newcommand*{\glsSetSuffixFF}[1]{%
519   \renewcommand*{\gls@suffixFF}{#1}%
520 }
```

`\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”.

```
521 \ifcsundef{hyperlink}%
522 {%
523   \newcommand*\glsnumberformat[1]{#1}%
524 }%
525 {%
526   \newcommand*\glsnumberformat[1]{\glshypernumber{#1}}%
527 }
```



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`

```
528 \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`

```
529 \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`

```
530 \newcommand*{\glossarypreamble}{}%
```

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`

```
531 \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`.

```
532 \newcommand*{\glossarysection}[2][\@gls@title]{%
533   \def\@gls@title{#2}%
534   \ifcsundef{phantomsection}%
```

```

535 {%
536   \@glossarysection{#1}{#2}%
537 }%
538 {%
539   \@pglossarysection{#1}{#2}%
540 }%
541 \glossarymark{\glossarytoctitle}%
542 }

```

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

```

543 \ifcsundef{glossarymark}%
544 {%
545   \ifglsucmark
546     \newcommand{\glossarymark}[1]{%
547       \@mkboth{\MakeUppercase{#1}}{\MakeUppercase{#1}}%
548     }
549   \else
550     \newcommand{\glossarymark}[1]{\@mkboth{#1}{#1}}
551   \fi
552 }%
553 {%
554   \GlossariesWarning{overriding \string\glossarymark}%
555   \ifclassloaded{memoir}%
556   {
557     \ifglsucmark
558       \renewcommand{\glossarymark}[1]{%
559         \@mkboth{\MakeUppercase{#1}}{\MakeUppercase{#1}}%
560       }
561     \else
562       \renewcommand{\glossarymark}[1]{%
563         \markboth{\memUHead{#1}}{\memUHead{#1}}%
564       }
565     \fi
566   }
567   {
568     \ifglsucmark
569       \renewcommand{\glossarymark}[1]{%
570         \@mkboth{\MakeUppercase{#1}}{\MakeUppercase{#1}}%
571       }
572     \else
573       \renewcommand{\glossarymark}[1]{\@mkboth{#1}{#1}}
574     \fi
575   }
576 }

```

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`

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

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

`\@glossarysection`

```
579 \newcommand*\@glossarysection[2]{%
580 \ifx\@glossarysecstar\@empty
581   \csname\@glossarysec\endcsname{#2}%
582 \else
583   \csname\@glossarysec\endcsname*{#2}%
584   \gls@toc{#1}{\@glossarysec}%
585 \fi
586 \@glossaryseclabel}
```

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`

```
587 \newcommand*\@pglossarysection[2]{%
588 \glsclearpage
589 \phantomsection
590 \ifx\@glossarysecstar\@empty
591   \csname\@glossarysec\endcsname{#2}%
592 \else
593   \gls@toc{#1}{\@glossarysec}%
594   \csname\@glossarysec\endcsname*{#2}%
595 \fi
596 \@glossaryseclabel}
```

`\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.

```
597 \newcommand*\gls@doclearpage{%
598   \ifthenelse{\equal{\@glossarysec}{chapter}}{
599     {%
600       \ifcsundef{cleardoublepage}{\clearpage}{\cleardoublepage}%
601     }%
602   }%
603 }
```

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

```
604 \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`

```
605 \newcommand*\@gls@toc[2]{%
606 \ifglstoc
607   \ifglslnumberline
608     \addcontentsline{toc}{#2}{\numberline{#1}}%
609   \else
610     \addcontentsline{toc}{#2}{#1}%
611   \fi
612 }
```

## 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

```
613 \newcommand*\glsnoxindywarning[1]{%
614   \GlossariesWarning{Not in xindy mode --- ignoring \string#1}%
615 }
```

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

```
616 \ifglsxindy
617   \edef\@xdyattributes{\string"default\string"}%
618 \fi
```

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

```
619 \ifglsxindy
620   \edef\@xdyattributelist{}%
621 \fi
```

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

```
622 \ifglsxindy
623   \def\@xdylocref{}
624 \fi
```

\@gls@ifinlist

```

625 \newcommand*{\@gls@ifinlist}[4]{%
626   \def\@do@ifinlist##1,#1,##2\end@do@ifinlist{%
627     \def\@gls@listsuffix{##2}%
628     \ifx\@gls@listsuffix\@empty
629       #4%
630     \else
631       #3%
632     \fi
633   }%
634   \@do@ifinlist,#2,#1,\end@do@ifinlist
635 }

```

\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.

```

636 \ifglxsindy
637   \newcommand*{\@xdycounters}{\@glscounter}
638   \newcommand*\GlsAddXdyCounters[1]{%
639     \@for\@gls@ctr:=#1\do{%

```

Check if already in list before adding.

```

640       \edef\@do@addcounter{%
641         \noexpand\@gls@ifinlist{\@gls@ctr}{\@xdycounters}{}%
642         {%
643           \noexpand\edef\noexpand\@xdycounters{\@xdycounters,%
644             \noexpand\@gls@ctr}%
645         }%
646       }%
647       \@do@addcounter
648     }
649   }

```

Only has an effect before \writeist:

```

650   \@onlypremakeg\GlsAddXdyCounters
651 \else
652   \newcommand*\GlsAddXdyCounters[1]{%
653     \glsnxindywarning\GlsAddXdyAttribute
654   }
655 \fi

```

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

```

656 \newcommand*\@disabled@glssaddxdycounters{%
657   \PackageError{glossaries}{\string\GlsAddXdyCounters\space
658     can't be used after \string\GlsAddXdyAttribute}{Move all
659     occurrences of \string\GlsAddXdyCounters\space before the first
660     instance of \string\GlsAddXdyAttribute}%
661 }

```

`\GlsAddXdyAttribute` Adds an attribute.

```

662 \ifglxindy
    First define internal command that adds an attribute for a given counter (2nd
    argument is the counter):
663   \newcommand*\@glsaddxdyattribute[2]{%
    Add to xindy attribute list
664     \edef\@xdyattributes{\@xdyattributes ^^J \string"#1\string" ^^J
665       \string"#2#1\string"}%
    Add to xindy markup location.
666     \expandafter\toks@\expandafter{\@xdylocref}%
667     \edef\@xdylocref{\the\toks@ ^^J%
668       (markup-locref
669       :open \string"\string~n%
670         \expandafter\string\csname glsX#2X#1\endcsname
671         \string" ^^J
672       :close \string"\string" ^^J
673       :attr \string"#2#1\string")}%
    Define associated attribute command \glsX<counter>X<attribute>{\<Hprefix>}{\<n>}
674     \expandafter\gdef\csname glsX#2X#1\endcsname##1##2{%
675       \setentrycounter[##1]{#2}\csname #1\endcsname{##2}%
676     }%
677   }

    High-level command:
678   \newcommand*\GlsAddXdyAttribute[1]{%
    Add to comma-separated attribute list
679     \ifx\@xdyattributelist\@empty
680       \edef\@xdyattributelist{#1}%
681     \else
682       \edef\@xdyattributelist{\@xdyattributelist,#1}%
683     \fi

    Iterate through all specified counters and add counter-dependent attributes:
684     \@for\@this@counter:=\@xdycounters\do{%
685       \protected@edef\gls@do@addxdyattribute{%
686         \noexpand\@glsaddxdyattribute{#1}{\@this@counter}%
687       }
688       \gls@do@addxdyattribute
689     }%

    All occurrences of \GlsAddXdyCounters must be used before this command
690     \let\GlsAddXdyCounters\@disabled@glsaddxdycounters
691   }

    Only has an effect before \writeist:
692   \@onlypremakeg\GlsAddXdyAttribute
693 \else
694   \newcommand*\GlsAddXdyAttribute[1]{%

```

```

695 \glsnoxywarning\GlsAddXdyAttribute}
696 \fi

```

redefinedattributes    Add known attributes for all defined counters

```

697 \ifglxindy
698 \newcommand*{\@gls@addpredefinedattributes}{%
699   \GlsAddXdyAttribute{glsnumberformat}
700   \GlsAddXdyAttribute{textrm}
701   \GlsAddXdyAttribute{textsf}
702   \GlsAddXdyAttribute{texttt}
703   \GlsAddXdyAttribute{textbf}
704   \GlsAddXdyAttribute{textmd}
705   \GlsAddXdyAttribute{textit}
706   \GlsAddXdyAttribute{textup}
707   \GlsAddXdyAttribute{textsl}
708   \GlsAddXdyAttribute{textsc}
709   \GlsAddXdyAttribute{emph}
710   \GlsAddXdyAttribute{glshypernumber}
711   \GlsAddXdyAttribute{hyperrm}
712   \GlsAddXdyAttribute{hypersf}
713   \GlsAddXdyAttribute{hypertt}
714   \GlsAddXdyAttribute{hyperbf}
715   \GlsAddXdyAttribute{hypermd}
716   \GlsAddXdyAttribute{hyperit}
717   \GlsAddXdyAttribute{hyperup}
718   \GlsAddXdyAttribute{hypersl}
719   \GlsAddXdyAttribute{hypersc}
720   \GlsAddXdyAttribute{hyperemph}
721 }
722 \else
723   \let\@gls@addpredefinedattributes\relax
724 \fi

```

\@xdyuseralphabets    List of additional alphabets

```

725 \def\@xdyuseralphabets{}

```

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

```

726 \ifglxindy
727   \newcommand*{\GlsAddXdyAlphabet}[2]{%
728     \edef\@xdyuseralphabets{%
729       \@xdyuseralphabets ^^J
730       (define-alphabet "#1" (#2))}%
731   \else
732     \newcommand*{\GlsAddXdyAlphabet}[2]{%
733       \glsnoxywarning\GlsAddXdyAlphabet}
734 \fi

```

This code is only required for xindy:

735 \ifglxindy

ls@xdy@locationlist List of predefined location names.

```
736 \newcommand*{\@gls@xdy@locationlist}{%
737     roman-page-numbers,%
738     Roman-page-numbers,%
739     arabic-page-numbers,%
740     alpha-page-numbers,%
741     Alpha-page-numbers,%
742     Appendix-page-numbers,%
743     arabic-section-numbers%
744 }
```

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.

```
745 \protected@edef\@gls@roman{\@roman{0}\string"
746     \string"roman-numbers-lowercase\string" :sep \string"}}%
747 \@onelevel@sanitize\@gls@roman
748 \edef\@tmp{\string" \string"roman-numbers-lowercase\string"
749     :sep \string"}%
750 \@onelevel@sanitize\@tmp
751 \ifx\@tmp\@gls@roman
752     \expandafter
753     \edef\csname @gls@xdy@Lclass@roman-page-numbers\endcsname{%
754         \string"roman-numbers-lowercase\string"%
755     }%
756 \else
757     \expandafter
758     \edef\csname @gls@xdy@Lclass@roman-page-numbers\endcsname{
759         :sep \string"\@gls@roman\string"%
760     }%
761 \fi
```

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

```
762 \expandafter\def\csname @gls@xdy@Lclass@Roman-page-numbers\endcsname{%
763     \string"roman-numbers-uppercase\string"%
764 }
```

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

```
765 \expandafter\def\csname @gls@xdy@Lclass@arabic-page-numbers\endcsname{%
766     \string"arabic-numbers\string"%
767 }
```

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



```

768 \expandafter\def\csname @gls@xdy@Lclass@alpha-page-numbers\endcsname{%
769 \string"alpha\string"%
770 }%

@Alpha-page-numbers Upper case alphabetical (A, B, ...).
771 \expandafter\def\csname @gls@xdy@Lclass@Alpha-page-numbers\endcsname{%
772 \string"ALPHA\string"%
773 }%

appendix-page-numbers Appendix style locations (e.g. A-1, A-2, ..., B-1, B-2, ...). The separator is given
by \@glsAlphacompositor.
774 \expandafter\def\csname @gls@xdy@Lclass@Appendix-page-numbers\endcsname{%
775 \string"ALPHA\string"
776 :sep \string"\@glsAlphacompositor\string"
777 \string"arabic-numbers\string"%
778 }

arabic-section-numbers Section number style locations (e.g. 1.1, 1.2, ...). The compositor is given by
\glscompositor.
779 \expandafter\def\csname @gls@xdy@Lclass@arabic-section-numbers\endcsname{%
780 \string"arabic-numbers\string"
781 :sep \string"\glscompositor\string"
782 \string"arabic-numbers\string"%
783 }%

xdyuserlocationdefs List of additional location definitions (separated by ^^J)
784 \def\@xdyuserlocationdefs{}

xdyuserlocationnames List of additional user location names
785 \def\@xdyuserlocationnames{}

End of xindy-only block:
786 \fi

\GlsAddXdyLocation \GlsAddXdyLocation[<prefix-loc>]{<name>}{<definition>} Define a new lo-
cation 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.)
787 \ifglsxindy
788 \newcommand*\GlsAddXdyLocation[3][{}]{%
789 \def\@gls@tmp{#1}%
790 \ifx\@gls@tmp\@empty
791 \edef\@xdyuserlocationdefs{%
792 \@xdyuserlocationdefs ^^J%
793 (define-location-class \string"#2\string"^^J\space\space
794 \space(:sep \string"{}\glsopenbrace\string" #3
795 :sep \string"\glsclosebrace\string"))

```

```

796     }%
797   \else
798     \edef\@xdyuserlocationdefs{%
799       \@xdyuserlocationdefs ^^J%
800       (define-location-class \string"#2\string"^^J\space\space
801       \space(:sep "\glssopenbrace"
802         #1
803         :sep "\glsclosebrace\glssopenbrace" #3
804         :sep "\glsclosebrace"))
805     }%
806   \fi
807   \edef\@xdyuserlocationnames{%
808     \@xdyuserlocationnames^^J\space\space\space
809     \string"#1\string"}%
810 }

```

Only has an effect before \writeist:

```

811 \@onlypremakeg\GlsAddXdyLocation
812 \else
813   \newcommand*\GlsAddXdyLocation}[2]{%
814     \glsnnoxindywarning\GlsAddXdyLocation}
815 \fi

```

ylocationclassorder Define location class order

```

816 \ifglxindy
817   \edef\@xdylocationclassorder{^^J\space\space\space
818     \string"roman-page-numbers\string"^^J\space\space\space
819     \string"arabic-page-numbers\string"^^J\space\space\space
820     \string"arabic-section-numbers\string"^^J\space\space\space
821     \string"alpha-page-numbers\string"^^J\space\space\space
822     \string"Roman-page-numbers\string"^^J\space\space\space
823     \string"Alpha-page-numbers\string"^^J\space\space\space
824     \string"Appendix-page-numbers\string"
825     \@xdyuserlocationnames^^J\space\space\space
826     \string"see\string"
827   }
828 \fi

```

Change the location order.

yLocationClassOrder

```

829 \ifglxindy
830   \newcommand*\GlsSetXdyLocationClassOrder[1]{%
831     \def\@xdylocationclassorder{#1}}
832 \else
833   \newcommand*\GlsSetXdyLocationClassOrder[1]{%
834     \glsnnoxindywarning\GlsSetXdyLocationClassOrder}
835 \fi

```

\@xdysortrules Define sort rules

```

836 \ifglxindy
837   \def\@xdysortrules{}
838 \fi

```

`\GlsAddSortRule`    Add a sort rule

```

839 \ifglxindy
840   \newcommand*\GlsAddSortRule[2]{%
841     \expandafter\toks@\expandafter{\@xdysortrules}%
842     \protected@edef\@xdysortrules{\the\toks@ ^^J
843       (sort-rule \string"#1\string" \string"#2\string"))}%
844   }
845 \else
846   \newcommand*\GlsAddSortRule[2]{%
847     \glsnxindywarning\GlsAddSortRule}
848 \fi

```

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

```

849 \ifglxindy
850   \def\@xdyrequiredstyles{tex}
851 \fi

```

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

```

852 \ifglxindy
853   \newcommand*\GlsAddXdyStyle[1]{%
854     \edef\@xdyrequiredstyles{\@xdyrequiredstyles,#1}}%
855 \else
856   \newcommand*\GlsAddXdyStyle[1]{%
857     \glsnxindywarning\GlsAddXdyStyle}
858 \fi

```

`\GlsSetXdyStyles`    Reset the list of required styles

```

859 \ifglxindy
860   \newcommand*\GlsSetXdyStyles[1]{%
861     \edef\@xdyrequiredstyles{#1}}
862 \else
863   \newcommand*\GlsSetXdyStyles[1]{%
864     \glsnxindywarning\GlsSetXdyStyles}
865 \fi

```

`\findrootlanguage`    The root language name is required by xindy. This information is for makeglossaries to pass to xindy. Since `\language` only stores the regional dialect rather than the root language name, some trickery is required to determine the root language.

```

866 \ifglxindy
867   \@ifpackageloaded{babel}{%

```

Need to parse babel.sty to determine the root language. This code was provided by Enrico Gregorio.

```

868 \def\findrootlanguage{\begingroup
869   \escapechar=-1\relax
   normalize \language to category 12 chars
870   \edef\language{\%
871     \expandafter\string\csname\language\endcsname}%
   disable babel.sty useless commands
872   \def\NeedsTeXFormat##1[##2]{}%
873   \def\ProvidesPackage##1[##2]{}%
874   \let\LdfInit\relax
875   \def\languageattribute##1##2{%
   change the meaning of \DeclareOption
876   \def\DeclareOption##1##2{%
   at \DeclareOption* we end
877   \ifx##1*\expandafter\endinput\else
   else we build a string with the first argument
878   \edef\testlanguage{\expandafter\string\csname##1\endcsname}%
   if \testlanguage and \language are the same we execute the second
   argument
879   \ifx\testlanguage\language##2\fi
880   \fi}
   almost all options of babel are \input{<name>.ldf}
881   \def\input##1{\stripldf##1}%
   we put the root language name in \rootlanguage
882   \def\stripldf##1.ldf{\gdef\rootlanguage{##1}}%
   now input babel.sty, using the primitive \input
883   \@@input babel.sty
884   \endgroup}%
885   }{%
   hasn't been loaded, so check if has been loaded
886   \@ifpackageloaded{ngerman}{%
887     \def\findrootlanguage{%
888       \def\rootlanguage{german}}%
889   }{%
   Neither babel nor ngerman have been loaded, so assume the root language is
   English
890   \def\findrootlanguage{%
891     \def\rootlanguage{english}}%
892   }%
893   }%
894 \fi

```

`\rootlanguage` Set default root language to English.

```
895 \def\rootlanguage{english}
```

`\@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.

```
896 \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.

```
897 \ifglxindy
898   \newcommand*\GlsSetXdyLanguage[2][\glsdefaultttype]{%
899   \ifglossaryexists{#1}{%
900     \expandafter\def\csname @xdy@#1@language\endcsname{#2}%
901   }{%
902     \PackageError{glossaries}{Can't set language type for
903     glossary type '#1' --- no such glossary}{%
904     You have specified a glossary type that doesn't exist}}
905 \else
906   \newcommand*\GlsSetXdyLanguage[2][]{%
907     \glsnoxywarning\GlsSetXdyLanguage}
908 \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.

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

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

```
910 \ifglxindy
911   \newcommand*\GlsSetXdyCodePage[1]{%
912     \renewcommand*\@gls@codepage{#1}%
913   }
914 \else
915   \newcommand*\GlsSetXdyCodePage[1]{%
916     \glsnoxywarning\GlsSetXdyCodePage}
917 \fi
```

`\@xdylettergroups` Store letter group definitions.

```
918 \ifglxindy
919   \ifglx@xindy@glsnumbers
920     \def\@xdylettergroups{(define-letter-group
921       \string"glsnumbers\string"^^J\space\space\space
922       :prefixes (\string"0\string" \string"1\string"
```

```

923     \string"2\string" \string"3\string" \string"4\string"
924     \string"5\string" \string"6\string" \string"7\string"
925     \string"8\string" \string"9\string")^^J\space\space\space
926     :before \string"\@glsfirstletter\string"))}
927 \else
928     \def\@xdylettergroups{}
929 \fi
930 \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.

```

931 \newcommand*\GlsAddLetterGroup[2]{%
932     \expandafter\toks@\expandafter{\@xdylettergroups}%
933     \protected@edef\@xdylettergroups{\the\toks@^^J%
934     (define-letter-group \string"#1\string"^^J\space\space\space#2)}}%
935 }%

```

## 1.5 Loops and conditionals

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

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

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

```

936 \newcommand*\forallglossaries[3][\@glo@types]{%
937     \@for#2:=#1\do{\ifx#2\@empty\else#3\fi}%
938 }

```

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

```
\forglentries[<type>]{<cmd>}{<code>}
```

where *<type>* is the glossary label and *<cmd>* is a control sequence which will be set to the entry label in the current iteration.

```

939 \newcommand*\forglentries[3][\glsdefaulttype]{%
940     \edef\@glo@list{\csname glolist@#1\endcsname}%
941     \@for#2:=\@glo@list\do{\ifx#2\@empty\else#3\fi}%
942 }

```

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

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

Within **\forallglentries**, the current glossary type is given by **\@this@glo@**.

```

943 \newcommand*\forallglentries[3][\@glo@types]{%

```

```

944 \expandafter\forallglossaries\expandafter[#1]{\@@this@glo@}{%
945 \forallglsentries[\@@this@glo@]{#2}{#3}}

```

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

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

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

```

946 \newcommand{\ifglossaryexists}[3]{%
947   \ifcsundef{glo@#1@out}{#3}{#2}%
948 }

```

`\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.

```

949 \newcommand{\ifglentryexists}[3]{%
950   \ifcsundef{glo@#1@name}{#3}{#2}%
951 }

```

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

```
\ifglsused{<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>`.

```
952 \newcommand*\ifglsused[3]{\ifthenelse{\boolean{glo@#1@flag}}{#2}{#3}}
```

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

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

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

```

953 \newcommand{\glsdoifexists}[2]{%
954   \ifglentryexists{#1}{#2}{%
955     \PackageError{glossaries}{Glossary entry ‘#1’ has not been
956     defined}{You need to define a glossary entry before you
957     can use it.}}%
958 }

```

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

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

```

959 \newcommand{\glsdoifnoexists}[2]{%
960   \ifglentryexists{#1}{%
961     \PackageError{glossaries}{Glossary entry ‘#1’ has already

```

```

962   been defined}}{#2}%
963 }

```

```

\ifglshaschildren \ifglshaschildren{<label>}{<true part>}{<false part>}
964 \newcommand{\ifglshaschildren}[3]{%
965   \glsdoifexists{#1}%
966   {%
967     \def\do@glshaschildren{#3}%
968     \expandafter\for@gl@sentries\expandafter[\csname glo@#1@type\endcsname]
969     {\glo@label}%
970     {%
971       \letcs\glo@parent{\glo@\glo@label @parent}%
972       \ifthenelse{\equal{#1}{\glo@parent}}{%
973         {%
974           \def\do@glshaschildren{#2}%
975           \@endfortrue
976         }%
977       }%
978     }%
979     \do@glshaschildren
980   }%
981 }

```

```

\ifglshasparent \ifglshaschildren{<label>}{<true part>}{<false part>}
982 \newcommand{\ifglshasparent}[3]{%
983   \glsdoifexists{#1}%
984   {%
985     \ifcsempy{\glo@#1@parent}{#3}{#2}%
986   }%
987 }

```

## 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
988 \newcommand*{\@glo@types}{,}

```

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`

```
989 \newcommand*{\newglossary}[5][glg]{%
990 \ifglossaryexists{#2}{%
991   \PackageError{glossaries}{Glossary type ‘#2’ already exists}{%
992     You can’t define a new glossary called ‘#2’ because it already
993     exists}%
994 }{%
```

Check if default has been set

```
995   \ifx\glsdefaulttype\relax
996     \gdef\glsdefaulttype{#2}%
997   \fi
```

Add this to the list of glossary types:

```
998   \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.

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

Store details of this new glossary type:

```
1000   \expandafter\def\csname @glotype@#2@in\endcsname{#3}%
1001   \expandafter\def\csname @glotype@#2@out\endcsname{#4}%
1002   \expandafter\def\csname @glotype@#2@title\endcsname{#5}%
1003   \protected@write\@auxout{}{\string\@newglossary{#2}{#1}{#3}{#4}}%
```

How to display this entry in the document text (uses `\glsdisplay` and `\glsdisplayfirst` by default). These can be redefined by the user later if required (see `\defglsdisplay` and `\defglsdisplayfirst`). These may already have been defined if this has been specified as a list of acronyms.

```
1004   \ifcsundef{gls@#2@display}%
1005   {%
1006     \expandafter\gdef\csname gls@#2@display\endcsname{\glsdisplay}%
1007   }%
1008   {%
1009     \ifcsundef{gls@#2@displayfirst}%
1010     {%
1011       \expandafter\gdef\csname gls@#2@displayfirst\endcsname{%
1012         \glsdisplayfirst
1013       }%
1014     }%
1015   }%
```

Define sort counter if required:

```
1016 \@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.)

```
1017 \@ifnextchar[{\@gls@setcounter{#2}}%
1018   {\@gls@setcounter{#2}[\glscounter]}}%
1019 }
```

`\altnewglossary`

```
1020 \newcommand*{\altnewglossary}[3]{%
1021   \newglossary[#2-glg]{#1}{#2-gls}{#2-glo}{#3}%
1022 }
```

Only define new glossaries in the preamble:

```
1023 \@onlypreamble{\newglossary}
```

Only define new glossaries before `\makeglossaries`

```
1024 \@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`

```
1025 \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`

```
1026 \def\@gls@setcounter#1[#2]{%
1027   \expandafter\def\csname @glotype@#1@counter\endcsname{#2}%
```

Add counter to xindy list, if not already added:

```
1028   \ifglxindy
1029     \GlsAddXdyCounters{#2}%
1030   \fi
1031 }
```

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

`\@gls@getcounter`

```
1032 \newcommand*{\@gls@getcounter}[1]{%
1033   \csname @glotype@#1@counter\endcsname}
```

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

```
1034 \glsdefmain
```

## 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.

```
1035 \define@key{glossentry}{name}{%
1036 \def\@glo@name{#1}%
1037 }
```

**description** The description key is usually only used in the glossary, but can be made to appear in the text by redefining `\glsdisplay` and `\glsdisplayfirst` (or using `\defglsdisplay` and `\defglsdisplayfirst`), however, you will have to disable the `sanitize` option (using the `sanitize` package option, `sanitize={description=false}`, and protect fragile commands). The description key is required when defining a new glossary entry. (Be careful not to make the description too long, because `makeindex` has a limited buffer. `\@glo@desc` is defined to be a short command to discourage lengthy descriptions for this reason. If you do have a very long description, or if you require paragraph breaks, define a separate command that contains the description, and use it as the value to the description key.)

```
1038 \define@key{glossentry}{description}{%
1039 \def\@glo@desc{#1}%
1040 }
```

**descriptionplural**

```
1041 \define@key{glossentry}{descriptionplural}{%
1042 \def\@glo@descplural{#1}%
1043 }
```

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

```
1044 \define@key{glossentry}{sort}{%
1045 \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.

```
1046 \define@key{glossentry}{text}{%
```

```
1047 \def\@glo@text{#1}%
1048 }
```

**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.

```
1049 \define@key{glossentry}{plural}{%
1050 \def\@glo@plural{#1}%
1051 }
```

**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.

```
1052 \define@key{glossentry}{first}{%
1053 \def\@glo@first{#1}%
1054 }
```

**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.

```
1055 \define@key{glossentry}{firstplural}{%
1056 \def\@glo@firstplural{#1}%
1057 }
```

**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 `\glossaryentryfield` so that it uses its fourth parameter. If you want this value to appear in the text when the term is used by commands like `\gls`, you will need to change `\glsdisplay` and `\glsdisplayfirst` (either explicitly for all glossaries or via `\defglsdisplay` and `\defglsdisplayfirst` for individual glossaries).

```
1058 \define@key{glossentry}{symbol}{%
1059 \def\@glo@symbol{#1}%
1060 }
```

**symbolplural**

```
1061 \define@key{glossentry}{symbolplural}{%
1062 \def\@glo@symbolplural{#1}%
1063 }
```

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

```
1064 \define@key{glossentry}{type}{%
1065 \def\@glo@type{#1}}
```

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

```
1066 \define@key{glossentry}{counter}{%
1067   \ifcsundef{c@#1}%
1068   {%
1069     \PackageError{glossaries}%
1070     {There is no counter called ‘#1’}%
1071     {%
1072       The counter key should have the name of a valid counter
1073       as its value%
1074     }%
1075   }%
1076   {%
1077     \def\@glo@counter{#1}%
1078   }%
1079 }
```

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

```
1080 \define@key{glossentry}{see}{%
1081   \def\@glo@see{#1}%
1082   \@glo@seeautonumberlist
1083 }
```

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

```
1084 \define@key{glossentry}{parent}{%
1085 \def\@glo@parent{#1}}
```

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

```
1086 \define@choicekey{glossentry}{nonumberlist}[\val\nr]{true,false}[true]{%
1087   \ifcase\nr\relax
1088     \def\@glo@prefix{\glsnonextpages}%
1089   \else
1090     \def\@glo@prefix{\glsnextpages}%
1091   \fi
1092 }
```

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

**user1**

```
1093 \define@key{glossentry}{user1}{%
1094   \def\@glo@useri{#1}%
1095 }
```

**user2**

```
1096 \define@key{glossentry}{user2}{%
1097   \def\@glo@userii{#1}%
1098 }
```

user3

```
1099 \define@key{glossentry}{user3}{%  
1100   \def\@glo@useriii{#1}%  
1101 }
```

user4

```
1102 \define@key{glossentry}{user4}{%  
1103   \def\@glo@useriv{#1}%  
1104 }
```

user5

```
1105 \define@key{glossentry}{user5}{%  
1106   \def\@glo@userv{#1}%  
1107 }
```

user6

```
1108 \define@key{glossentry}{user6}{%  
1109   \def\@glo@uservi{#1}%  
1110 }
```

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.

```
1111 \define@key{glossentry}{short}{%  
1112   \def\@glo@short{#1}%  
1113 }
```

shortplural This key is provided for use by \newacronym.

```
1114 \define@key{glossentry}{shortplural}{%  
1115   \def\@glo@shortpl{#1}%  
1116 }
```

long This key is provided for use by \newacronym.

```
1117 \define@key{glossentry}{long}{%  
1118   \def\@glo@long{#1}%  
1119 }
```

longplural This key is provided for use by \newacronym.

```
1120 \define@key{glossentry}{longplural}{%  
1121   \def\@glo@longpl{#1}%  
1122 }
```

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

```
1123 \newcommand*\@glsnname{%  
1124   \PackageError{glossaries}{name key required in  
1125   \string\newglossaryentry\space for entry '\@glo@label'}{You  
1126   haven't specified the entry name}}
```

`\@glsdefaultplural` Define command to set default plural.

```

1127 \newcommand*{\@glsdefaultplural}{\@glo@text\glspluralsuffix}

```

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

```

1128 \newcommand*{\@gls@missingnumberlist}[1]{%
1129   ??%
1130   \ifglssavenumberlist
1131     \GlossariesWarning{Missing number list for entry ‘#1’.
1132       Maybe makeglossaries + rerun required.}%
1133   \else
1134     \PackageError{glossaries}%
1135       {Package option ‘savenumberlist=true’ required.}%
1136     {%
1137       You must use the ‘savenumberlist’ package option
1138       to reference location lists.%
1139     }%
1140   \fi
1141 }

```

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

```

1142 \newcommand*{\@glsdefaultsort}{\@glo@name}

```

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

```

1143 \newcount\gls@level

```

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

```

1144 \newrobustcmd{\newglossaryentry}[2]{%

```

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

```

1145   \glsdoifnoexists{#1}%
1146   {%

```

Store label

```

1147     \def\@glo@label{#1}%

```

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

```

1148     \let\@glo@name\@glsnname

```

```

1149     \def\@glo@desc{%
1150       \PackageError{glossaries}
1151       {%
1152         description key required in \string\newglossaryentry\space
1153         for entry ‘\@glo@label’%
1154       }%
1155     {%
1156       You haven’t specified the entry description%
1157     }%
1158   }%

```

```

1159 \def\@glo@descplural{\@glo@desc}%
1160 \def\@glo@type{\glsdefaulttype}%
1161 \def\@glo@symbol{\relax}%

1162 \def\@glo@symbolplural{\@glo@symbol}%
1163 \def\@glo@text{\@glo@name}%
1164 \let\@glo@plural\@glsdefaultplural

```

Using `\let` instead of `\def` to make later comparison avoid expansion issues.  
(Thanks to Ulrich Diez for suggesting this.)

```

1165 \let\@glo@first\relax

1166 \let\@glo@firstplural\relax

```

Set the default sort:

```

1167 \let\@glo@sort\@glsdefaultsort

```

Set the default counter:

```

1168 \def\@glo@counter{\@gls@getcounter{\@glo@type}}%
1169 \def\@glo@see{}%

1170 \def\@glo@parent{}%

1171 \def\@glo@prefix{}%

1172 \def\@glo@useri{}%
1173 \def\@glo@userii{}%
1174 \def\@glo@useriii{}%
1175 \def\@glo@useriv{}%
1176 \def\@glo@userv{}%
1177 \def\@glo@uservi{}%

1178 \def\@glo@short{}%
1179 \def\@glo@shortpl{}%
1180 \def\@glo@long{}%
1181 \def\@glo@longpl{}%

```

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

```

1182 \@newglossaryentryprehook

```

Extract key-val information from third parameter:

```

1183 \setkeys{glossentry}{#2}%

```

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

```

1184 \ifcsundef{glolist@\@glo@type}%
1185 {%
1186   \PackageError{glossaries}%
1187   {Glossary type '\@glo@type' has not been defined}%

```



```

1188      {You need to define a new glossary type, before making entries
1189      in it}%
1190    }%
1191    {%
1192      \protected@edef\@glolist@\csname glo\list@\@glo@type\endcsname}%
1193      \expandafter\xdef\csname glo\list@\@glo@type\endcsname{\@glolist@{#1},}%
1194    }%

  Initialise level to 0.
1195    \gls@level=0\relax

  Has this entry been assigned a parent?
1196    \ifx\@glo@parent\@empty

    Doesn't have a parent. Set \glo@<label>@parent to empty.
1197      \expandafter\gdef\csname glo@#1@parent\endcsname{}%
1198    \else

    Has a parent. Check to ensure this entry isn't its own parent.
1199      \ifthenelse{\equal{#1}{\@glo@parent}}{%
1200        {%
1201          \PackageError{glossaries}{Entry '#1' can't be its own parent}{}%
1202          \def\@glo@parent{}%
1203          \expandafter\gdef\csname glo@#1@parent\endcsname{}%
1204        }%
1205      }%

    Check the parent exists:
1206      \ifglsentryexists{\@glo@parent}%
1207      {%

    Parent exists. Set \glo@<label>@parent.
1208      \expandafter\xdef\csname glo@#1@parent\endcsname{\@glo@parent}%

    Determine level.
1209      \gls@level=\csname glo@\@glo@parent @level\endcsname\relax
1210      \advance\gls@level by 1\relax

    If name hasn't been specified, use same as the parent name
1211      \ifx\@glo@name\@glsnoname
1212        \expandafter\let\expandafter\@glo@name
1213        \csname glo@\@glo@parent @name\endcsname

    If name and plural haven't been specified, use same as the parent
1214      \ifx\@glo@plural\@glsdefaultplural
1215        \expandafter\let\expandafter\@glo@plural
1216        \csname glo@\@glo@parent @plural\endcsname
1217      \fi
1218    \fi
1219  }%
1220  {%

```

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

```

1221      \PackageError{glossaries}%
1222      {%
1223          Invalid parent '@glo@parent'
1224          for entry '#1' - parent doesn't exist%
1225      }%
1226      {%
1227          Parent entries must be defined before their children%
1228      }%
1229      \def@glo@parent{%
1230          \expandafter\gdef\csname glo@#1@parent\endcsname{%
1231      }%
1232      }%
1233      \fi

```

Set the level for this entry

```

1234      \expandafter\xdef\csname glo@#1@level\endcsname{\number\gls@level}%

```

Check if first and firstplural have been use. If firstplural hasn't been specified, but first has been specified, then form firstplural by appending \glspluralsuffix to value of first key, otherwise obtain the value from the plural key. This now uses \ifx instead of \if to avoid expansion issues. (Thanks to Ulrich Diez for suggesting this.)

```

1235      \ifx\relax@glo@firstplural
1236          \ifx\relax@glo@first
1237              \def@glo@firstplural{\@glo@plural}%
1238              \def@glo@first{\@glo@text}%
1239          \else
1240              \def@glo@firstplural{\@glo@first\glspluralsuffix}%
1241          \fi
1242      \else
1243          \ifx\relax@glo@first
1244              \def@glo@first{\@glo@text}%
1245          \fi
1246      \fi

```

Define commands associated with this entry:

```

1247      \expandafter
1248      \protected\xdef\csname glo@#1@text\endcsname{\@glo@text}%
1249      \expandafter
1250      \protected\xdef\csname glo@#1@plural\endcsname{\@glo@plural}%
1251      \expandafter
1252      \protected\xdef\csname glo@#1@first\endcsname{\@glo@first}%
1253      \expandafter
1254      \protected\xdef\csname glo@#1@firstpl\endcsname{\@glo@firstplural}%
1255      \expandafter
1256      \protected\xdef\csname glo@#1@type\endcsname{\@glo@type}%
1257      \expandafter
1258      \protected\xdef\csname glo@#1@counter\endcsname{\@glo@counter}%

```

```

1259 \expandafter
1260 \protected@xdef\csname glo@#1@useri\endcsname{\@glo@useri}%
1261 \expandafter
1262 \protected@xdef\csname glo@#1@userii\endcsname{\@glo@userii}%
1263 \expandafter
1264 \protected@xdef\csname glo@#1@useriii\endcsname{\@glo@useriii}%
1265 \expandafter
1266 \protected@xdef\csname glo@#1@useriv\endcsname{\@glo@useriv}%
1267 \expandafter
1268 \protected@xdef\csname glo@#1@userv\endcsname{\@glo@userv}%
1269 \expandafter
1270 \protected@xdef\csname glo@#1@uservi\endcsname{\@glo@uservi}%
1271 \expandafter
1272 \protected@xdef\csname glo@#1@short\endcsname{\@glo@short}%
1273 \expandafter
1274 \protected@xdef\csname glo@#1@shortpl\endcsname{\@glo@shortpl}%
1275 \expandafter
1276 \protected@xdef\csname glo@#1@long\endcsname{\@glo@long}%
1277 \expandafter
1278 \protected@xdef\csname glo@#1@longpl\endcsname{\@glo@longpl}%
1279 \@gls@sanitizename
1280 \expandafter\protected@xdef\csname glo@#1@name\endcsname{\@glo@name}%

```

Set default numberlist if not defined:

```

1281 \ifcsundef{glo@#1@numberlist}%
1282 {%
1283 \csxdef{glo@#1@numberlist}{\noexpand\@gls@missingnumberlist{\@glo@label}}%
1284 }%
1285 {}%

```

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.

```

1286 \def\@glo@@desc{\@glo@first}%
1287 \ifx\@glo@desc\@glo@@desc
1288 \let\@glo@desc\@glo@first
1289 \fi
1290 \@gls@sanitizedesc
1291 \expandafter\protected@xdef\csname glo@#1@desc\endcsname{\@glo@desc}%
1292 \expandafter\protected@xdef\csname glo@#1@descplural\endcsname{\@glo@descplural}%

```

Set the sort key for this entry:

```

1293 \@gls@defsort{\@glo@type}{#1}%

1294 \def\@glo@@symbol{\@glo@text}%
1295 \ifx\@glo@symbol\@glo@@symbol
1296 \let\@glo@symbol\@glo@text
1297 \fi
1298 \@gls@sanitizesymbol
1299 \expandafter\protected@xdef\csname glo@#1@symbol\endcsname{\@glo@symbol}%
1300 \expandafter\protected@xdef\csname glo@#1@symbolplural\endcsname{\@glo@symbolplural}%

```

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

```

1301 \expandafter\gdef\csname glo@#1@flagfalse\endcsname{%
1302 \expandafter\global\expandafter
1303 \let\csname ifglo@#1@flag\endcsname\iffalse
1304 }%
1305 \expandafter\gdef\csname glo@#1@flagtrue\endcsname{%
1306 \expandafter\global\expandafter
1307 \let\csname ifglo@#1@flag\endcsname\iftrue
1308 }%
1309 \csname glo@#1@flagfalse\endcsname

```

Sort out any cross-referencing if required.

```

1310 \ifx\@glo@see\@empty
1311 \else
1312 \protected@edef\@do@glsee{%
1313 \noexpand\@gl@fixbraces\noexpand\@glo@list\@glo@see
1314 \noexpand\@nil
1315 \noexpand\expandafter\noexpand\@glsee\noexpand\@glo@list{#1}}%
1316 \@do@glsee
1317 \fi
1318 }%

```

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

```

1319 \do@glo@storeentry{#1}%

```

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

```

1320 \@newglossaryentryposthook
1321 }

```

**glossaryentryprehook** Allow extra information to be added to glossary entries:

```

1322 \newcommand*{\@newglossaryentryprehook}{}

```

**glossaryentryposthook** Allow extra information to be added to glossary entries:

```

1323 \newcommand*{\@newglossaryentryposthook}{}

```

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

```

1324 \newcommand*{\glsmoveentry}[2]{%
1325 \edef\glo@type{\csname glo@#1@type\endcsname}%
1326 \def\glo@list{,%}
1327 \forglsentries[\glo@type]{\glo@label}%
1328 {%
1329 \ifthenelse{\equal{\glo@label}{#1}}{\eappto\glo@list{\glo@label,}}%
1330 }%
1331 \cslet{glolist@\glo@type}{\glo@list}%
1332 \csdef{glo@#1@type}{#2}%
1333 }

```

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

```
1334 \ifglxindy
1335   \newcommand*{\@glossaryentryfield}{\string\@glossaryentryfield}
1336 \else
1337   \newcommand*{\@glossaryentryfield}{\string\glossaryentryfield}
1338 \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.)

```
1339 \ifglxindy
1340   \newcommand*{\@glossarysubentryfield}{%
1341     \string\@glossarysubentryfield}
1342 \else
1343   \newcommand*{\@glossarysubentryfield}{%
1344     \string\glossarysubentryfield}
1345 \fi
```

`\@glo@storeentry` Determine the format to write the entry in the glossary output (`.glo`) file. The argument is the entry's label. The result is stored in `\glo@<label>@entry`, where `<label>` is the entry's label. (This doesn't include any formatting or location information.)

```
1346 \newcommand{\@glo@storeentry}[1]{%
  Get the sort string and escape any special characters
1347 \protected@edef\@glo@sort{\csname glo@#1@sort\endcsname}%
1348 \@gls@checkmkidxchars\@glo@sort
  Same again for the name string.
1349 \protected@edef\@glo@name{\csname glo@#1@name\endcsname}%
1350 \@gls@checkmkidxchars\@glo@name
  Add the font command. (The backslash needs to be escaped for xindy.)
1351 \ifglxindy
1352   \protected@edef\@glo@name{\string\@gls@namefont{\@glo@name}}%
1353 \else
1354   \protected@edef\@glo@name{\string\gls@namefont{\@glo@name}}%
1355 \fi
  Get the description string and escape any special characters
1356 \protected@edef\@glo@desc{\csname glo@#1@desc\endcsname}%
1357 \@gls@checkmkidxchars\@glo@desc
  Same again for the symbol
1358 \protected@edef\@glo@symbol{\csname glo@#1@symbol\endcsname}%
1359 \@gls@checkmkidxchars\@glo@symbol
  Escape any special characters in the prefix
1360 \@gls@checkmkidxchars\@glo@prefix
```

Get the parent, if one exists

```
1361 \edef\@glo@parent{\csname glo@#1@parent\endcsname}%
```

Write the information to the glossary file.

```
1362 \ifglxindy
```

Store using xindy syntax.

```
1363 \ifx\@glo@parent\@empty
```

Entry doesn't have a parent

```
1364 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
1365 (\string"\@glo@sort\string" %
1366 \string"\@glo@prefix\@glossaryentryfield{#1}\@glo@name
1367 }\@glo@desc}\@glo@symbol}\string") %
1368 }%
1369 \else
```

Entry has a parent

```
1370 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
1371 \csname glo@\@glo@parent @index\endcsname
1372 (\string"\@glo@sort\string" %
1373 \string"\@glo@prefix\@glossarysubentryfield%
1374 {\csname glo@#1@level\endcsname}{#1}\@glo@name
1375 }\@glo@desc}\@glo@symbol}\string") %
1376 }%
1377 \fi
1378 \else
```

Store using makeindex syntax.

```
1379 \ifx\@glo@parent\@empty
```

Sanitize \@glo@prefix

```
1380 \@onelevel@sanitize\@glo@prefix
```

Entry doesn't have a parent

```
1381 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
1382 \@glo@sort\@gls@actualchar\@glo@prefix
1383 \@glossaryentryfield{#1}\@glo@name}\@glo@desc
1384 }\@glo@symbol}%
1385 }%
1386 \else
```

Entry has a parent

```
1387 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
1388 \csname glo@\@glo@parent @index\endcsname\@gls@levelchar
1389 \@glo@sort\@gls@actualchar\@glo@prefix
1390 \@glossarysubentryfield
1391 {\csname glo@#1@level\endcsname}{#1}\@glo@name}\@glo@desc
1392 }\@glo@symbol}%
1393 }%
1394 \fi
1395 \fi
1396 }
```

## 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:

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.

`\glsreset`

```
1397 \newcommand*{\glsreset}[1]{%
1398 \glsdoifexists{#1}{%
1399 \expandafter\global\csname glo@#1@flagfalse\endcsname}}
```

As above, but with only a local effect:

`\glslocalreset`

```
1400 \newcommand*{\glslocalreset}[1]{%
1401 \glsdoifexists{#1}{%
1402 \expandafter\let\csname ifglo@#1@flag\endcsname\iffalse}}
```

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.

`\glsunset`

```
1403 \newcommand*{\glsunset}[1]{%
1404 \glsdoifexists{#1}{%
1405 \expandafter\global\csname glo@#1@flagtrue\endcsname}}
```

As above, but with only a local effect:

`\glslocalunset`

```
1406 \newcommand*{\glslocalunset}[1]{%
1407 \glsdoifexists{#1}{%
1408 \expandafter\let\csname ifglo@#1@flag\endcsname\iftrue}}
```

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

`\glsresetall`

```
1409 \newcommand*{\glsresetall}[1][\@glo@types]{%
1410 \forallglsentries[1]{\@glsentry}{%
1411 \glsreset{\@glsentry}}}
```

As above, but with only a local effect:

`\glslocalresetall`

```
1412 \newcommand*{\glslocalresetall}[1][\@glo@types]{%
1413 \forallglsentries[1]{\@glsentry}{%
1414 \glslocalreset{\@glsentry}}}
```

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

```
\glsunsetall
1415 \newcommand*{\glsunsetall}[1][\@gls@types]{%
1416 \forallglsentries[#1]{\@glsentry}{%
1417 \glsunset{\@glsentry}}}
```

As above, but with only a local effect:

```
\glslocalunsetall
1418 \newcommand*{\glslocalunsetall}[1][\@gls@types]{%
1419 \forallglsentries[#1]{\@glsentry}{%
1420 \glslocalunset{\@glsentry}}}
```

## 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
1421 \newcommand*{\loadglsentries}[2][\@gls@default]{%
1422 \let\@gls@default\glsdefaulttype
1423 \def\glsdefaulttype{#1}\input{#2}%
1424 \let\glsdefaulttype\@gls@default}
```

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

```
1425 \@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”.

---

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



`\glstextformat`

```
1426 \newcommand*{\glstextformat}[1]{#1}
```

The first time an entry is used, the way in which it is displayed is governed by `\glsdisplayfirst`. This takes four parameters: #1 will be the value of the entry's first or firstplural key, #2 will be the value of the entry's description key, #3 will be the value of the entry's symbol key and #4 is additional text supplied by the final optional argument to commands like `\gls` and `\glspl`. The default is to display the first parameter followed by the additional text.

`\glsdisplayfirst`

```
1427 \newcommand*{\glsdisplayfirst}[4]{#1#4}
```

After the first use, the entry is displayed according to the format of `\glsdisplay`. Again, it takes four parameters: #1 will be the value of the entry's text or plural key, #2 will be the value of the entry's description key, #3 will be the value of the entry's symbol key and #4 is additional text supplied by the final optional argument to commands like `\gls` and `\glspl`.

`\glsdisplay`

```
1428 \newcommand*{\glsdisplay}[4]{#1#4}
```

When a new glossary is created it uses `\glsdisplayfirst` and `\glsdisplay` as the default way of displaying its entry in the text. This can be changed for the entries belonging to an individual glossary using `\defglsdisplay` and `\defglsdisplayfirst`.

```
\defglsdisplay[⟨type⟩]{⟨definition⟩}
```

The glossary type is given by `⟨type⟩` (the default glossary if omitted) and `⟨definition⟩` should have at most #1, #2, #3 and #4. These represent the same arguments as those described for `\glsdisplay`.

`\defglsdisplay`

```
1429 \newcommand*{\defglsdisplay}[2][\glsdefaulttype]{%
1430 \expandafter\def\csname gls@#1@display\endcsname##1##2##3##4{#2}}
```

```
\defglsdisplayfirst[⟨type⟩]{⟨definition⟩}
```

The glossary type is given by `⟨type⟩` (the default glossary if omitted) and `⟨definition⟩` should have at most #1, #2, #3 and #4. These represent the same arguments as those described for `\glsdisplayfirst`.

`\defglsdisplayfirst`

```
1431 \newcommand*{\defglsdisplayfirst}[2][\glsdefaulttype]{%
1432 \expandafter\def\csname gls@#1@displayfirst\endcsname##1##2##3##4{#2}}
```

### 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`, 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 `\defglsdisplay` and `\defglsdisplayfirst`). 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.

```
1433 \define@key{glslink}{counter}{%
1434   \ifcsundef{c@#1}%
1435   {%
1436     \PackageError{glossaries}%
1437     {There is no counter called ‘#1’}%
1438     {%
1439       The counter key should have the name of a valid counter
1440       as its value%
1441     }%
1442   }%
1443   {%
1444     \def\@gls@counter{#1}%
1445   }%
1446 }
```

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.

```
1447 \define@key{glslink}{format}{%
1448 \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.

```
1449 \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.

```
1450 \define@boolkey{glslink}{local}[true]{}
```

Syntax:

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

Display  $\langle text \rangle$  in the document, and add the entry information for  $\langle label \rangle$  into the relevant glossary. The optional argument should be a key value list using the `glslink` keys defined above.

There is also a starred version:

`\glslink*[\langle options \rangle]{\langle label \rangle}{\langle text \rangle}`

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

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

```
\glslink
1451 \newrobustcmd*{\glslink}{%
1452 \@ifstar\@sgls@link\@gls@link}

\@sgls@link The starred version of \glslink calls the unstarred version with hyperlinks dis-
abled.
1453 \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.
1454 \newcommand*{\@gls@link}[3] [] {%
1455   \ifglsentryexists{#2}%
1456   {%
1457     \@gls@link[#1]{#2}{#3}%
1458   }{%
1459     \PackageError{glossaries}{Glossary entry '#2' has not been
1460     defined}{You need to define a glossary entry before you
1461     can use it.}%
1462     \glsformat{#3}%
1463   }%
1464 }

\@gls@link
1465 \def\@gls@link[#1]#2#3{%
Inserting \leavevmode suggested by Donald Arseneau (avoids problem with
tabularx).
1466   \leavevmode
1467   \def\glslabel{#2}%
1468   \def\glsnumberformat{\glsnumberformat}%
1469   \edef\@gls@counter{\csname glo@#2@counter\endcsname}%
1470   \KV@glslink@hypertrue
1471   \setkeys{glslink}{#1}%
Store the entry's counter in \theglsentrycounter
1472   \@gls@saveentrycounter
```

Define sort key if necessary:

```

1473 \gls@setsort{#2}%

1474 \do@wrglossary{#2}%
1475 \ifKV@glslink@hyper
1476 \glslink{\glolinkprefix#2}{\glstextformat{#3}}%
1477 \else
1478 \glstextformat{#3}\relax
1479 \fi
1480 }

```

\glolinkprefix

```

1481 \newcommand*{\glolinkprefix}{glo:}

```

\glsentrycounter Set default value of entry counter

```

1482 \def\glsentrycounter{\glscounter}%

```

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

```

1483 \newcommand*{\gls@saveentrycounter}{%
1484 \def\gls@Hcounter{}%

```

Are we using equation counter?

```

1485 \ifthenelse{\equal{\gls@counter}{equation}}{%
1486 {

```

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

```

1487 \ifcsundef{xatlevel@}%
1488 {%
1489 \edef\theglsentrycounter{\expandafter\noexpand
1490 \csname the\gls@counter\endcsname}%
1491 }%
1492 {%
1493 \ifx\xatlevel@\@empty
1494 \edef\theglsentrycounter{\expandafter\noexpand
1495 \csname the\gls@counter\endcsname}%
1496 \else
1497 \savecounters@
1498 \advance\c@equation by 1\relax
1499 \edef\theglsentrycounter{\csname the\gls@counter\endcsname}%

```

Check if hyperref version of this counter

```

1500 \ifcsundef{theH\gls@counter}%
1501 {%
1502 \def\gls@Hcounter{\theglsentrycounter}%
1503 }%
1504 {%
1505 \def\gls@Hcounter{\csname theH\gls@counter\endcsname}%
1506 }%
1507 \protected@edef\theHglentrycounter{\gls@Hcounter}%

```

```

1508      \restorecounters@
1509      \fi
1510    }%
1511  }%
1512  {%

```

Not using equation counter so no special measures:

```

1513      \edef\theglsentrycounter{\expandafter\noexpand
1514      \csname the\@gls@counter\endcsname}%
1515    }%

```

Check if hyperref version of this counter

```

1516    \ifx\@gls@Hcounter\@empty
1517      \ifcsundef{theH\@gls@counter}%
1518        {%
1519          \def\theHglentrycounter{\theglsentrycounter}%
1520        }%
1521        {%
1522          \protected@edef\theHglentrycounter{\expandafter\noexpand
1523          \csname theH\@gls@counter\endcsname}%
1524        }%
1525    \fi
1526 }

```

`\@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.

```

1527 \def\@set@glo@numformat#1#2#3#4{%
1528   \expandafter\@glo@check@mkidxrangechar#3\@nil
1529   \protected@edef#1{%
1530     \@glo@prefix setentrycounter[#4]{#2}%
1531     \expandafter\string\csname\@glo@suffix\endcsname
1532   }%
1533   \@gls@checkmkidxchars#1%
1534 }

```

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.

```

1535 \def\@glo@check@mkidxrangechar#1#2\@nil{%
1536   \if#1(\relax
1537     \def\@glo@prefix{(%
1538     \if\relax#2\relax
1539       \def\@glo@suffix{glsnumberformat}%
1540     \else

```

```

1541 \def\@glo@suffix{#2}%
1542 \fi
1543 \else
1544 \if#1)\relax
1545 \def\@glo@prefix{}}%
1546 \if\relax#2\relax
1547 \def\@glo@suffix{glsnumberformat}%
1548 \else
1549 \def\@glo@suffix{#2}%
1550 \fi
1551 \else
1552 \def\@glo@prefix{}\def\@glo@suffix{#1#2}%
1553 \fi
1554 \fi}

```

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

```

1555 \newcommand*{\@gls@escbsdq}[1]{%
1556 \def\@gls@checkedmkidx{}%
1557 \let\gls@xdystring=#1\relax
1558 \@onelevel@sanitize\gls@xdystring
1559 \edef\do@gls@xdycheckbackslash{%
1560 \noexpand\@gls@xdycheckbackslash\gls@xdystring\noexpand\@nil
1561 \@backslashchar\@backslashchar\noexpand\null}%
1562 \do@gls@xdycheckbackslash
1563 \expandafter\@gls@updatechecked\@gls@checkedmkidx{\gls@xdystring}%
1564 \def\@gls@checkedmkidx{}%
1565 \expandafter\@gls@xdycheckquote\gls@xdystring\@nil""\null
1566 \expandafter\@gls@updatechecked\@gls@checkedmkidx{\gls@xdystring}%

```

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

```

1567 \@for\@gls@tmp:=\gls@protected@pagefmts\do
1568 {%
1569 \edef\@gls@sanitized@tmp{\expandafter\@gobble\string\\expandonce\@gls@tmp}%
1570 \@onelevel@sanitize\@gls@sanitized@tmp
1571 \edef\gls@dosubst{%
1572 \noexpand\DTLsubstituteall\noexpand\gls@xdystring
1573 {\@gls@sanitized@tmp}{\expandonce\@gls@tmp}%
1574 }%
1575 \gls@dosubst
1576 }%

```

Assign to required control sequence

```

1577 \let#1=\gls@xdystring
1578 }

```

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

`\gls@checkmkidxchars`

```

1579 \newcommand{\@gls@checkmkidxchars}[1]{%
1580 \ifglxsindy
1581   \@gls@escbsdq{#1}%
1582 \else
1583   \def\@gls@checkedmkidx{}%
1584   \expandafter\@gls@checkquote#1\@nil""\null
1585   \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
1586   \def\@gls@checkedmkidx{}%
1587   \expandafter\@gls@checkescquote#1\@nil\""\null
1588   \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
1589   \def\@gls@checkedmkidx{}%
1590   \expandafter\@gls@checkescactual#1\@nil\?\?\null
1591   \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
1592   \def\@gls@checkedmkidx{}%
1593   \expandafter\@gls@checkactual#1\@nil??\null
1594   \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
1595   \def\@gls@checkedmkidx{}%
1596   \expandafter\@gls@checkbar#1\@nil||\null
1597   \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
1598   \def\@gls@checkedmkidx{}%
1599   \expandafter\@gls@checkeschar#1\@nil|||\null
1600   \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
1601   \def\@gls@checkedmkidx{}%
1602   \expandafter\@gls@checklevel#1\@nil!!\null
1603   \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
1604 \fi
1605 }

```

Update the control sequence and strip trailing \@nil:

\@gls@updatechecked

```

1606 \def\@gls@updatechecked#1\@nil#2{\def#2{#1}}

```

\@gls@tmpb    Define temporary token

```

1607 \newtoks\@gls@tmpb

```

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

```

1608 \def\@gls@checkquote#1"#2"#3\null{%
1609 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1610 \toks@={#1}%
1611 \ifx\null#2\null
1612 \ifx\null#3\null
1613 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1614 \def\@gls@checkquote{\relax}%
1615 \else
1616 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1617   \@gls@quotechar\@gls@quotechar\@gls@quotechar\@gls@quotechar}%
1618 \def\@gls@checkquote{\@gls@checkquote#3\null}%
1619 \fi

```

```

1620 \else
1621 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1622 \@gls@quotechar\@gls@quotechar}%
1623 \ifx\@null#3\@null
1624 \def\@gls@checkquote{\@gls@checkquote#2""\@null}%
1625 \else
1626 \def\@gls@checkquote{\@gls@checkquote#2"#3\@null}%
1627 \fi
1628 \fi
1629 \@gls@checkquote}

```

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

```

1630 \def\@gls@checkescquote#1\"#2\"#3\@null{%
1631 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1632 \toks@={#1}%
1633 \ifx\@null#2\@null
1634 \ifx\@null#3\@null
1635 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1636 \def\@gls@checkescquote{\relax}%
1637 \else
1638 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1639 \@gls@quotechar\string\" \@gls@quotechar
1640 \@gls@quotechar\string\" \@gls@quotechar}%
1641 \def\@gls@checkescquote{\@gls@checkescquote#3\@null}%
1642 \fi
1643 \else
1644 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1645 \@gls@quotechar\string\" \@gls@quotechar}%
1646 \ifx\@null#3\@null
1647 \def\@gls@checkescquote{\@gls@checkescquote#2\"\" \@null}%
1648 \else
1649 \def\@gls@checkescquote{\@gls@checkescquote#2\"#3\@null}%
1650 \fi
1651 \fi
1652 \@gls@checkescquote}

```

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

```

1653 \def\@gls@checkescactual#1\"#2\"#3\@null{%
1654 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1655 \toks@={#1}%
1656 \ifx\@null#2\@null
1657 \ifx\@null#3\@null
1658 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1659 \def\@gls@checkescactual{\relax}%
1660 \else
1661 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1662 \@gls@quotechar\string\" \@gls@actualchar
1663 \@gls@quotechar\string\" \@gls@actualchar}%
1664 \def\@gls@checkescactual{\@gls@checkescactual#3\@null}%

```



```

1665 \fi
1666 \else
1667 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1668   \@gls@quotechar\string"\@gls@actualchar}%
1669 \ifx\null#3\null
1670   \def\@gls@checkescactual{\@gls@checkescactual#2\?\?\null}%
1671 \else
1672   \def\@gls@checkescactual{\@gls@checkescactual#2\?#3\null}%
1673 \fi
1674 \fi
1675 \@gls@checkescactual}

```

\@gls@checkescbar Similarly for \|:

```

1676 \def\@gls@checkescbar#1\|#2\|#3\null{%
1677 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1678 \toks@={#1}%
1679 \ifx\null#2\null
1680   \ifx\null#3\null
1681     \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1682     \def\@gls@checkescbar{\relax}%
1683   \else
1684     \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1685       \@gls@quotechar\string"\@gls@encapchar
1686       \@gls@quotechar\string"\@gls@encapchar}%
1687     \def\@gls@checkescbar{\@gls@checkescbar#3\null}%
1688   \fi
1689 \else
1690   \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1691     \@gls@quotechar\string"\@gls@encapchar}%
1692   \ifx\null#3\null
1693     \def\@gls@checkescbar{\@gls@checkescbar#2\|\|\null}%
1694   \else
1695     \def\@gls@checkescbar{\@gls@checkescbar#2\|#3\null}%
1696   \fi
1697 \fi
1698 \@gls@checkescbar}

```

\@gls@checkesclevel Similarly for \!:

```

1699 \def\@gls@checkesclevel#1\!#2\!#3\null{%
1700 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1701 \toks@={#1}%
1702 \ifx\null#2\null
1703   \ifx\null#3\null
1704     \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1705     \def\@gls@checkesclevel{\relax}%
1706   \else
1707     \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1708       \@gls@quotechar\string"\@gls@levelchar
1709       \@gls@quotechar\string"\@gls@levelchar}%

```

```

1710 \def\@gls@checkesclevel{\@gls@checkesclevel#3\null}%
1711 \fi
1712 \else
1713 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1714 \@gls@quotechar\string"\@gls@levelchar}%
1715 \ifx\null#3\null
1716 \def\@gls@checkesclevel{\@gls@checkesclevel#2\!\!\null}%
1717 \else
1718 \def\@gls@checkesclevel{\@gls@checkesclevel#2\!#3\null}%
1719 \fi
1720 \fi
1721 \@gls@checkesclevel}

```

\@gls@checkbar and for |:

```

1722 \def\@gls@checkbar#1|#2|#3\null{%
1723 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1724 \toks@={#1}%
1725 \ifx\null#2\null
1726 \ifx\null#3\null
1727 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1728 \def\@gls@checkbar{\relax}%
1729 \else
1730 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1731 \@gls@quotechar\@gls@encapchar\@gls@quotechar\@gls@encapchar}%
1732 \def\@gls@checkbar{\@gls@checkbar#3\null}%
1733 \fi
1734 \else
1735 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1736 \@gls@quotechar\@gls@encapchar}%
1737 \ifx\null#3\null
1738 \def\@gls@checkbar{\@gls@checkbar#2||\null}%
1739 \else
1740 \def\@gls@checkbar{\@gls@checkbar#2|#3\null}%
1741 \fi
1742 \fi
1743 \@gls@checkbar}

```

\@gls@checklevel and for !:

```

1744 \def\@gls@checklevel#1!#2!#3\null{%
1745 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1746 \toks@={#1}%
1747 \ifx\null#2\null
1748 \ifx\null#3\null
1749 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1750 \def\@gls@checklevel{\relax}%
1751 \else
1752 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1753 \@gls@quotechar\@gls@levelchar\@gls@quotechar\@gls@levelchar}%
1754 \def\@gls@checklevel{\@gls@checklevel#3\null}%

```

```

1755 \fi
1756 \else
1757 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1758   \@gls@quotechar\@gls@levelchar}%
1759 \ifx\@null#3\@null
1760   \def\@gls@checklevel{\@gls@checklevel#2!!\@null}%
1761 \else
1762   \def\@gls@checklevel{\@gls@checklevel#2!#3\@null}%
1763 \fi
1764 \fi
1765 \@gls@checklevel}

```

\@gls@checkactual and for ?:

```

1766 \def\@gls@checkactual#1?#2?#3\@null{%
1767 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1768 \toks@={#1}%
1769 \ifx\@null#2\@null
1770 \ifx\@null#3\@null
1771   \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1772   \def\@gls@checkactual{\relax}%
1773 \else
1774   \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1775     \@gls@quotechar\@gls@actualchar\@gls@quotechar\@gls@actualchar}%
1776   \def\@gls@checkactual{\@gls@checkactual#3\@null}%
1777 \fi
1778 \else
1779 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1780   \@gls@quotechar\@gls@actualchar}%
1781 \ifx\@null#3\@null
1782   \def\@gls@checkactual{\@gls@checkactual#2??\@null}%
1783 \else
1784   \def\@gls@checkactual{\@gls@checkactual#2?#3\@null}%
1785 \fi
1786 \fi
1787 \@gls@checkactual}

```

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

```

1788 \def\@gls@xdycheckquote#1"#2"#3\@null{%
1789 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1790 \toks@={#1}%
1791 \ifx\@null#2\@null
1792 \ifx\@null#3\@null
1793   \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1794   \def\@gls@xdycheckquote{\relax}%
1795 \else
1796   \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1797     \string"\string"}%
1798   \def\@gls@xdycheckquote{\@gls@xdycheckquote#3\@null}%
1799 \fi

```

```

1800 \else
1801 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1802   \string\}%
1803 \ifx\null#3\null
1804   \def\@gls@xdycheckquote{\@gls@xdycheckquote#2""\null}%
1805 \else
1806   \def\@gls@xdycheckquote{\@gls@xdycheckquote#2"#3\null}%
1807 \fi
1808 \fi
1809 \@gls@xdycheckquote
1810 }

```

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

```

\@gls@xdycheckbackslash
1811 \edef\def@gls@xdycheckbackslash{%
1812   \noexpand\def\noexpand\@gls@xdycheckbackslash##1\@backslashchar
1813     ##2\@backslashchar##3\noexpand\null{%
1814     \noexpand\@gls@tmpb=\noexpand\expandafter
1815       {\noexpand\@gls@checkedmkidx}%
1816     \noexpand\toks@={##1}%
1817     \noexpand\ifx\noexpand\null##2\noexpand\null
1818       \noexpand\ifx\noexpand\null##3\noexpand\null
1819       \noexpand\edef\noexpand\@gls@checkedmkidx{%
1820         \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@}%
1821       \noexpand\def\noexpand\@gls@xdycheckbackslash{\relax}%
1822     \noexpand\else
1823       \noexpand\edef\noexpand\@gls@checkedmkidx{%
1824         \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
1825         \@backslashchar\@backslashchar\@backslashchar\@backslashchar}%
1826     \noexpand\def\noexpand\@gls@xdycheckbackslash{%
1827       \noexpand\@gls@xdycheckbackslash##3\noexpand\null}%
1828     \noexpand\fi
1829   \noexpand\else
1830     \noexpand\edef\noexpand\@gls@checkedmkidx{%
1831       \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
1832       \@backslashchar\@backslashchar}%
1833   \noexpand\ifx\noexpand\null##3\noexpand\null
1834     \noexpand\def\noexpand\@gls@xdycheckbackslash{%
1835       \noexpand\@gls@xdycheckbackslash##2\@backslashchar
1836       \@backslashchar\noexpand\null}%
1837   \noexpand\else
1838     \noexpand\def\noexpand\@gls@xdycheckbackslash{%
1839       \noexpand\@gls@xdycheckbackslash##2\@backslashchar
1840       ##3\noexpand\null}%
1841   \noexpand\fi
1842 \noexpand\fi
1843 \noexpand\@gls@xdycheckbackslash
1844 }%
1845 }

```

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

```
1846 \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`.

```
1847 \ifcsundef{hyperlink}%
1848 {%
1849   \gdef\@glslink#1#2{#2}%
1850 }%
1851 {%
1852   \gdef\@glslink#1#2{\hyperlink{#1}{#2}}%
1853 }
```

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

```
1854 \newlength\gls@tmplen
1855 \ifcsundef{hypertarget}%
1856 {%
1857   \gdef\@glstarget#1#2{#2}%
1858 }%
1859 {%
1860   \gdef\@glstarget#1#2{%
1861     \settoheight{\gls@tmplen}{#2}%
1862     \raisebox{\gls@tmplen}{\hypertarget{#1}{}}#2%
1863   }%
1864 }
```

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

`\glsdisablehyper`

```
1865 \newcommand{\glsdisablehyper}{%
1866 \renewcommand*\@glslink[2]{##2}%
1867 \renewcommand*\@glstarget[2]{##2}}
```

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

`\glsenablehyper`

```
1868 \newcommand{\glsenablehyper}{%
1869 \renewcommand*\@glslink[2]{\hyperlink{##1}{##2}}%
1870 \renewcommand*\@glstarget[2]{%
1871   \settoheight{\gls@tmplen}{##2}%
1872   \raisebox{\gls@tmplen}{\hypertarget{##1}{}}##2}}
```

Syntax:

`\gls[⟨options⟩]{⟨label⟩}[⟨insert text⟩]`

Link to glossary entry using singular form. The link text is taken from the value of the text or first keys used when the entry was defined.

The first optional argument is a key-value list, the same as `\glslink`, the mandatory argument is the entry label. After the mandatory argument, there is another optional argument to insert extra text in the link text (the location of the inserted text is governed by `\glsdisplay` and `\glsdisplayfirst`). As with `\glslink` there is a starred version which is the same as the unstarred version but with the hyper key set to false. (Additional options can also be specified in the first optional argument.)

First determine if we are using the starred form:

`\gls`

```
1873 \newrobustcmd*{\gls}{\@ifstar\@sgls\@gls}
```

Define the starred form:

`\@sgls`

```
1874 \newcommand*{\@sgls}[1] []{\@gls[hyper=false,#1]}
```

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

`\@gls`

```
1875 \newcommand*{\@gls}[2] []{%
1876   \new@ifnextchar[{\@gls@{#1}{#2}}{\@gls@{#1}{#2} []}%
1877 }
```

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

```
1878 \def\@gls@#1#2[#3]{%
1879   \glsdoifexists{#2}%
1880   {%
1881     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
1882   \def\@gls@link@opts{#1}%
1883   \def\@gls@link@label{#2}%
```

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

```
1884   \ifglsused{#2}%
1885   {%
1886     \def\@glo@text{%
1887       \csname gls@\@glo@type @display\endcsname
1888       {\glsentrytext{#2}}{\glsentrydesc{#2}}{\glsentrysymbol{#2}}{#3}}%
1889   }%
1890   {%
1891     \def\@glo@text{%
1892       \csname gls@\@glo@type @displayfirst\endcsname
1893       {\glsentryfirst{#2}}{\glsentrydesc{#2}}{\glsentrysymbol{#2}}{#3}}%
1894   }%
```

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.

```

1895   \ifglsused{#2}%
1896   {%
1897     \@gls@link[#1]{#2}{\@glo@text}%
1898   }%
1899   {%
1900     \gls@checkisacronymlist\@glo@type
1901     \ifthenelse
1902       {(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote})}
1903       {OR \NOT\boolean{gls hyperfirst}}
1904     }%
1905     {%
1906       \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
1907     }%
1908     {%
1909       \@gls@link[#1]{#2}{\@glo@text}%
1910     }%
1911   }%

```

Indicate that this entry has now been used

```

1912   \ifKV@glslink@local
1913     \glslocalunset{#2}%
1914   \else
1915     \glsunset{#2}%
1916   \fi
1917 }%
1918 }

```

`\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`

```

1919 \newrobustcmd*{\Gls}{\@ifstar\@sGls\@Gls}

```

Define the starred form:

```

1920 \newcommand*{\@sGls}[1] [] {\@Gls[hyper=false,#1]}

```

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

```

1921 \newcommand*{\@Gls}[2] [] {%
1922   \new@ifnextchar[{\@Gls@{#1}{#2}}{\@Gls@{#1}{#2} []}%
1923 }

```

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

```

1924 \def\@Gls@#1#2[#3]{%

```

```

1925 \glsdoifexists{#2}%
1926 {%
1927 \edef\@glo@type{\glsentrytype{#2}}%
    Save options in \@gls@link@opts and label in \@gls@link@label
1928 \def\@gls@link@opts{#1}%
1929 \def\@gls@link@label{#2}%
1930 \def\glslabel{#2}%
    Determine what the link text should be (this is stored in \@glo@text)
1931 \ifglsused{#2}%
1932 {%
1933 \protected@edef\@glo@text{%
1934 \csname gls@\@glo@type @display\endcsname
1935 {\glsentrytext{#2}}{\glsentrydesc{#2}}%
1936 {\glsentrysymbol{#2}}{#3}}%
1937 }%
1938 {%
1939 \protected@edef\@glo@text{%
1940 \csname gls@\@glo@type @displayfirst\endcsname
1941 {\glsentryfirst{#2}}{\glsentrydesc{#2}}%
1942 {\glsentrysymbol{#2}}{#3}}%
1943 }%
    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.
1944 \ifglsused{#2}%
1945 {%
1946 \@gls@link[#1]{#2}{%
1947 \expandafter\makefirstuc\expandafter{\@glo@text}}%
1948 }%
1949 {%
1950 \gls@checkisacronymlist\@glo@type
1951 \ifthenelse
1952 {%
1953 \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
1954 \OR \NOT\boolean{glshyperfirst}}%
1955 }%
1956 {%
1957 \@gls@link[#1,hyper=false]{#2}{%
1958 \expandafter\makefirstuc\expandafter{\@glo@text}}%
1959 }%
1960 {%
1961 \@gls@link[#1]{#2}{%
1962 \expandafter\makefirstuc\expandafter{\@glo@text}}%
1963 }%
1964 }%
    Indicate that this entry has now been used
1965 \ifKV@glslink@local

```



```

1966     \glslocalunset{#2}%
1967     \else
1968     \glsunset{#2}%
1969     \fi
1970 }%
1971 }

```

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

\GLS

```

1972 \newrobustcmd*{\GLS}{\@ifstar\@sGLS\@GLS}

```

Define the starred form:

```

1973 \newcommand*{\@sGLS}[1][\@GLS[hyper=false,#1]]{

```

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

```

1974 \newcommand*{\@GLS}[2][\@GLS@#1]{%
1975   \new@ifnextchar[\@GLS@{#1}{#2}}{\@GLS@{#1}{#2}[]}%
1976 }

```

\@GLS@ Read in the final optional argument:

```

1977 \def\@GLS@#1#2[#3]{%
1978   \glsdoifexists{#2}%
1979   {%
1980     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

1981   \def\@gls@link@opts{#1}%
1982   \def\@gls@link@label{#2}%

```

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

```

1983   \ifglsused{#2}%
1984   {%
1985     \def\@glo@text{%
1986       \csname gls@\@glo@type @display\endcsname
1987       {\glsentrytext{#2}}{\glsentrydesc{#2}}{\glsentrysymbol{#2}}{#3}%
1988     }%
1989   }%
1990   {%
1991     \def\@glo@text{%
1992       \csname gls@\@glo@type @displayfirst\endcsname
1993       {\glsentryfirst{#2}}{\glsentrydesc{#2}}{\glsentrysymbol{#2}}{#3}%
1994     }%
1995   }%

```

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.

```

1996   \ifglsused{#2}%
1997   {%

```

```

1998     \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
1999   }%
2000   {%
2001     \gls@checkisacronymlist\@glo@type
2002     \ifthenelse
2003     {%
2004       \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
2005       \OR \NOT\boolean{glshyperfirst}}{%
2006       \@gls@link[#1,hyper=false]{#2}{\MakeUppercase{\@glo@text}}%
2007     }%
2008     {%
2009       \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
2010     }%
2011   }%

```

Indicate that this entry has now been used

```

2012     \ifKV@glslink@local
2013     \glslocalunset{#2}%
2014     \else
2015     \glsunset{#2}%
2016     \fi
2017   }%
2018 }

```

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

`\glspl`

```

2019 \newrobustcmd*{\glspl}{\@ifstar\@sglspl\@glspl}

```

Define the starred form:

```

2020 \newcommand*{\@sglspl}[1][\@glspl[hyper=false,#1]]

```

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

```

2021 \newcommand*{\@glspl}[2][\@glspl@{#1}{#2}]{\@glspl@{#1}{#2}[]}
2022 \new@ifnextchar[\@glspl@{#1}{#2}]{\@glspl@{#1}{#2}[]}
2023 }

```

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

```

2024 \def\@glspl@#1#2[#3]{%
2025   \glsdoifexists{#2}%
2026   {%
2027     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2028   \def\@gls@link@opts{#1}%
2029   \def\@gls@link@label{#2}%

```

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

```

2030   \ifglsused{#2}%
2031   {%

```

```

2032     \def\@glo@text{%
2033         \csname gls@\@glo@type @display\endcsname
2034         {\glsentryplural{#2}}{\glsentrydescplural{#2}}%
2035         {\glsentrysymbolplural{#2}}{#3}}%
2036     }%
2037     {%
2038     \def\@glo@text{%
2039         \csname gls@\@glo@type @displayfirst\endcsname
2040         {\glsentryfirstplural{#2}}{\glsentrydescplural{#2}}%
2041         {\glsentrysymbolplural{#2}}{#3}}%
2042     }%

```

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.

```

2043     \ifglused{#2}%
2044     {%
2045         \@gls@link[#1]{#2}{\@glo@text}%
2046     }%
2047     {%
2048         \gls@checkisacronymlist\@glo@type
2049         \ifthenelse
2050         {%
2051             \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
2052             \OR \NOT\boolean{glshyperfirst}}%
2053         }%
2054         {%
2055             \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
2056         }%
2057         {%
2058             \@gls@link[#1]{#2}{\@glo@text}%
2059         }%
2060     }%

```

Indicate that this entry has now been used

```

2061     \ifKV@glslink@local
2062     \glslocalunset{#2}%
2063     \else
2064     \glsunset{#2}%
2065     \fi
2066 }%
2067 }

```

`\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`

```

2068 \newrobustcmd*{\Glspl}{\@ifstar\@sGlspl\@Glspl}

```

Define the starred form:

```
2069 \newcommand*{\@Glspl}[1][\@Glspl[hyper=false,#1]]
```

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

```
2070 \newcommand*{\@Glspl}[2][\@Glspl[hyper=false,#1]]
```

```
2071 \new@ifnextchar[\@Glspl@{#1}{#2}]{\@Glspl@{#1}{#2}[]}%
```

```
2072 }
```

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

```
2073 \def\@Glspl@#1#2[#3]{%
```

```
2074 \glsdoifexists{#2}%
```

```
2075 {%
```

```
2076 \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2077 \def\@gls@link@opts{#1}%
```

```
2078 \def\@gls@link@label{#2}%
```

```
2079 \def\glslabel{#2}%
```

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`.

```
2080 \ifglsused{#2}%
```

```
2081 {%
```

```
2082 \protected@edef\@glo@text{%
```

```
2083 \csname gls@\@glo@type @display\endcsname
```

```
2084 {\glsentryplural{#2}}{\glsentrydescplural{#2}}%
```

```
2085 {\glsentrysymbolplural{#2}}{#3}}%
```

```
2086 }%
```

```
2087 {%
```

```
2088 \protected@edef\@glo@text{%
```

```
2089 \csname gls@\@glo@type @displayfirst\endcsname
```

```
2090 {\glsentryfirstplural{#2}}{\glsentrydescplural{#2}}%
```

```
2091 {\glsentrysymbolplural{#2}}{#3}}%
```

```
2092 }%
```

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.

```
2093 \ifglsused{#2}%
```

```
2094 {%
```

```
2095 \@gls@link[#1]{#2}{%
```

```
2096 \expandafter\xmakefirstuc\expandafter{\@glo@text}}%
```

```
2097 }%
```

```
2098 {%
```

```
2099 \gls@checkisacronymlist\@glo@type
```

```
2100 \ifthenelse
```

```
2101 {%
```

```
2102 \(\boolean{glsisacronymlist}\AND \boolean{glsacrfootnote})\)
```

```

2103      \OR \NOT\boolean{glshyperfirst}%
2104    }%
2105    {%
2106      \@gls@link[#1,hyper=false]{#2}{%
2107        \expandafter\makefirstuc\expandafter{\@glo@text}}%
2108      }%
2109      {%
2110        \@gls@link[#1]{#2}{%
2111          \expandafter\makefirstuc\expandafter{\@glo@text}}%
2112        }%
2113      }%

```

Indicate that this entry has now been used

```

2114    \ifKV@glslink@local
2115      \glslocalunset{#2}%
2116    \else
2117      \glsunset{#2}%
2118    \fi
2119  }%
2120 }

```

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

`\GLSp1`

```

2121 \newrobustcmd*{\GLSp1}{\@ifstar\@sGLSp1\@GLSp1}

```

Define the starred form:

```

2122 \newcommand*{\@sGLSp1}[1][\@GLSp1[hyper=false,#1]]

```

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

```

2123 \newcommand*{\@GLSp1}[2][\@GLSp1@{#1}{#2}]{\@GLSp1@{#1}{#2}[]}
2124 \new@ifnextchar[\@GLSp1@{#1}{#2}]{\@GLSp1@{#1}{#2}[]}
2125 }

```

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

```

2126 \def\@GLSp1@#1#2[#3]{%
2127   \glsdoifexists{#2}%
2128   {%
2129     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2130   \def\@gls@link@opts{#1}%
2131   \def\@gls@link@label{#2}%

```

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

```

2132   \ifglsused{#2}%
2133   {%
2134     \def\@glo@text{%
2135       \csname gls@\@glo@type @display\endcsname

```

```

2136      {\glsentryplural{#2}}{\glsentrydescplural{#2}}%
2137      {\glsentrysymbolplural{#2}}{#3}%
2138    }%
2139  }%
2140  {%
2141    \def\@glo@text{%
2142      \csname gls@\@glo@type @displayfirst\endcsname
2143      {\glsentryfirstplural{#2}}{\glsentrydescplural{#2}}%
2144      {\glsentrysymbolplural{#2}}{#3}%
2145    }%
2146  }%

```

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.

```

2147    \ifglsused{#2}%
2148    {%
2149      \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
2150    }%
2151    {%
2152      \gls@checkisacronymlist\@glo@type
2153      \ifthenelse
2154      {%
2155        \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
2156        \OR \NOT\boolean{glshyperfirst}%
2157      }%
2158      {%
2159        \@gls@link[#1,hyper=false]{#2}{\MakeUppercase{\@glo@text}}%
2160      }%
2161      {%
2162        \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
2163      }%
2164    }%

```

Indicate that this entry has now been used

```

2165    \ifKV@glslink@local
2166    \glslocalunset{#2}%
2167  \else
2168    \glsunset{#2}%
2169  \fi
2170 }%
2171 }

```

`\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:

```

2172 \newrobustcmd*{\glsdisp}{\@ifstar\@sglsdisp\@glsdisp}

```

Define the starred form:

\@sgls

```
2173 \newcommand*{\@sglsdisp}[1] [] {\@glsdisp[hyper=false,#1]}
```

Defined the un-starred form.

\@glsdisp

```
2174 \newcommand*{\@glsdisp}[3] [] {%
```

```
2175   \glsdoifexists{#2}{%
```

```
2176     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2177   \def\@gls@link@opts{#1}%
```

```
2178   \def\@gls@link@label{#2}%
```

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

```
2179   \ifglsused{#2}%
```

```
2180   {%
```

```
2181     \def\@glo@text{%
```

```
2182       \csname gls@\@glo@type @display\endcsname
```

```
2183       {#3}{\glsentrydesc{#2}}{\glsentrysymbol{#2}}{}}%
```

```
2184   }%
```

```
2185   {%
```

```
2186     \def\@glo@text{%
```

```
2187       \csname gls@\@glo@type @displayfirst\endcsname
```

```
2188       {#3}{\glsentrydesc{#2}}{\glsentrysymbol{#2}}{}}%
```

```
2189   }%
```

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.

```
2190   \ifglsused{#2}%
```

```
2191   {%
```

```
2192     \@gls@link[#1]{#2}{\@glo@text}%
```

```
2193   }%
```

```
2194   {%
```

```
2195     \gls@checkisacronymlist\@glo@type
```

```
2196     \ifthenelse{\(\boolean{@glsisacronymlist})\AND
```

```
2197       \boolean{glsacrfootnote}) \OR \NOT\boolean{glshyperfirst}}%
```

```
2198     {%
```

```
2199       \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
```

```
2200     }%
```

```
2201     {%
```

```
2202       \@gls@link[#1]{#2}{\@glo@text}%
```

```
2203     }%
```

```
2204   }%
```

Indicate that this entry has now been used

```
2205   \ifKV@glslink@local
```

```
2206   \glslocalunset{#2}%
```

```
2207   \else
```

```

2208     \glsunset{#2}%
2209     \fi
2210 }%
2211 }

```

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

`\gls`

```

2212 \newrobustcmd*{\gls}{\@ifstar\sgls\gls}

```

Define the starred form:

```

2213 \newcommand*{\sgls}[1] [] {\gls[hyper=false,#1]}

```

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

```

2214 \newcommand*{\gls}[2] [] {%
2215 \new@ifnextchar {\gls@{#1}{#2}}{\gls@{#1}{#2} []}}

```

Read in the final optional argument:

```

2216 \def\gls@{#1#2[#3]}{%
2217 \glsdoifexists{#2}{\edef\gls@type{\glsentrytype{#2}}}%

```

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

```

2218 \protected@edef\gls@text{\glsentrytext{#2}}%

```

Call `\gls@link`

```

2219 \gls@link[#1]{#2}{\gls@text#3}%
2220 }%
2221 }

```

`\GL` behaves like `\gls` except the text is converted to uppercase.

`\GL`

```

2222 \newrobustcmd*{\GL}{\@ifstar\sgl\GL}

```

Define the starred form:

```

2223 \newcommand*{\sgl}[1] [] {\GL[hyper=false,#1]}

```

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

```

2224 \newcommand*{\GL}[2] [] {%
2225 \new@ifnextchar {\GL@{#1}{#2}}{\GL@{#1}{#2} []}}

```

Read in the final optional argument:

```

2226 \def\GL@{#1#2[#3]}{%
2227 \glsdoifexists{#2}{\edef\gls@type{\glsentrytype{#2}}}%

```

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

```

2228 \protected@edef\gls@text{\glsentrytext{#2}}%

```



Call \@gls@link

```
2229 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%  
2230 }%  
2231 }
```

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

\Glstext

```
2232 \newrobustcmd*{\Glstext}{\@ifstar\@sGlstext\@Glstext}
```

Define the starred form:

```
2233 \newcommand*{\@sGlstext}[1][]{\@Glstext[hyper=false,#1]}
```

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

```
2234 \newcommand*{\@Glstext}[2][]{%  
2235 \new@ifnextchar[{\@Glstext@{#1}{#2}}{\@Glstext@{#1}{#2}[]}]}
```

Read in the final optional argument:

```
2236 \def\@Glstext@#1#2[#3]{%  
2237 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2238 \protected@edef\@glo@text{\glsentrytext{#2}}%
```

Call \@gls@link

```
2239 \@gls@link[#1]{#2}{%  
2240 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%  
2241 }%  
2242 }
```

\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

```
2243 \newrobustcmd*{\glsfirst}{\@ifstar\@sglsfirst\@glsfirst}
```

Define the starred form:

```
2244 \newcommand*{\@sglsfirst}[1][]{\@glsfirst[hyper=false,#1]}
```

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

```
2245 \newcommand*{\@glsfirst}[2][]{%  
2246 \new@ifnextchar[{\@glsfirst@{#1}{#2}}{\@glsfirst@{#1}{#2}[]}]}
```

Read in the final optional argument:

```
2247 \def\@glsfirst@#1#2[#3]{%  
2248 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2249 \protected@edef\@glo@text{\glsentryfirst{#2}}%
```

Call \@gls@link  
2250 \@gls@link[#1]{#2}{\@glo@text#3}%  
2251 }%  
2252 }

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

\Glsfirst

2253 \newrobustcmd\*{\Glsfirst}{\@ifstar\@sGlsfirst\@Glsfirst}

Define the starred form:

2254 \newcommand\*{\@sGlsfirst}[1][ ]{\@Glsfirst[hyper=false,#1]}

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

2255 \newcommand\*{\@Glsfirst}[2][ ]{%  
2256 \new@ifnextchar[{\@Glsfirst@{#1}{#2}}{\@Glsfirst@{#1}{#2}[]}]}

Read in the final optional argument:

2257 \def\@Glsfirst@#1#2[#3]{%  
2258 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

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

2259 \protected@edef\@glo@text{\glsentryfirst{#2}}%

Call \@gls@link

2260 \@gls@link[#1]{#2}{%  
2261 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%  
2262 }%  
2263 }

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

\GLSfirst

2264 \newrobustcmd\*{\GLSfirst}{\@ifstar\@sGLSfirst\@GLSfirst}

Define the starred form:

2265 \newcommand\*{\@sGLSfirst}[1][ ]{\@GLSfirst[hyper=false,#1]}

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

2266 \newcommand\*{\@GLSfirst}[2][ ]{%  
2267 \new@ifnextchar[{\@GLSfirst@{#1}{#2}}{\@GLSfirst@{#1}{#2}[]}]}

Read in the final optional argument:

2268 \def\@GLSfirst@#1#2[#3]{%  
2269 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

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

2270 \protected@edef\@glo@text{\glsentryfirst{#2}}%

Call \@gls@link

```
2271 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%  
2272 }%  
2273 }
```

\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

```
2274 \newrobustcmd*{\glsplural}{\@ifstar\@sglsplural\@glsplural}
```

Define the starred form:

```
2275 \newcommand*{\@sglsplural}[1][\@glsplural[hyper=false,#1]]
```

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

```
2276 \newcommand*{\@glsplural}[2][\@%  
2277 \new@ifnextchar[\@glsplural@{#1}{#2}]{\@glsplural@{#1}{#2}[]}]
```

Read in the final optional argument:

```
2278 \def\@glsplural@#1#2[#3]{%  
2279 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2280 \protected@edef\@glo@text{\glsentryplural{#2}}%
```

Call \@gls@link

```
2281 \@gls@link[#1]{#2}{\@glo@text#3}%  
2282 }%  
2283 }
```

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

\Glsplural

```
2284 \newrobustcmd*{\Glsplural}{\@ifstar\@sGlsplural\@Glsplural}
```

Define the starred form:

```
2285 \newcommand*{\@sGlsplural}[1][\@Glsplural[hyper=false,#1]]
```

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

```
2286 \newcommand*{\@Glsplural}[2][\@%  
2287 \new@ifnextchar[\@Glsplural@{#1}{#2}]{\@Glsplural@{#1}{#2}[]}]
```

Read in the final optional argument:

```
2288 \def\@Glsplural@#1#2[#3]{%  
2289 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2290 \protected@edef\@glo@text{\glsentryplural{#2}}%
```

Call \@gls@link

```

2291 \@gls@link[#1]{#2}{%
2292   \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2293 }%
2294 }

```

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

\GLSplural

```

2295 \newrobustcmd*{\GLSplural}{\@ifstar\@sGLSplural\@GLSplural}

```

Define the starred form:

```

2296 \newcommand*{\@sGLSplural}[1][\@GLSplural[hyper=false,#1]]{

```

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

```

2297 \newcommand*{\@GLSplural}[2][\@GLSplural@{#1}{#2}[]]{%
2298 \new@ifnextchar[\@GLSplural@{#1}{#2}]{\@GLSplural@{#1}{#2}[]}}

```

Read in the final optional argument:

```

2299 \def\@GLSplural@#1#2[#3]{%
2300 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2301 \protected@edef\@glo@text{\glsentryplural{#2}}%

```

Call \@gls@link

```

2302 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2303 }%
2304 }

```

\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

```

2305 \newrobustcmd*{\glsfirstplural}{\@ifstar\@sglsfirstplural\@glsfirstplural}

```

Define the starred form:

```

2306 \newcommand*{\@sglsfirstplural}[1][\@glsfirstplural[hyper=false,#1]]{

```

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

```

2307 \newcommand*{\@glsfirstplural}[2][\@glsfirstplural@{#1}{#2}[]]{%
2308 \new@ifnextchar[\@glsfirstplural@{#1}{#2}]{\@glsfirstplural@{#1}{#2}[]}}

```

Read in the final optional argument:

```

2309 \def\@glsfirstplural@#1#2[#3]{%
2310 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2311 \protected@edef\@glo@text{\glsentryfirstplural{#2}}%

```

Call \@gls@link

```
2312 \@gls@link[#1]{#2}{\@glo@text#3}%  
2313 }%  
2314 }
```

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

\Glsfirstplural

```
2315 \newrobustcmd*{\Glsfirstplural}{\@ifstar\@sGlsfirstplural\@Glsfirstplural}
```

Define the starred form:

```
2316 \newcommand*{\@sGlsfirstplural}[1][\@Glsfirstplural[hyper=false,#1]]
```

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

```
2317 \newcommand*{\@Glsfirstplural}[2][\@%  
2318 \new@ifnextchar[\@Glsfirstplural@{#1}{#2}]{\@Glsfirstplural@{#1}{#2}[]}]
```

Read in the final optional argument:

```
2319 \def\@Glsfirstplural@#1#2[#3]{%  
2320 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2321 \protected@edef\@glo@text{\glsentryfirstplural{#2}}%
```

Call \@gls@link

```
2322 \@gls@link[#1]{#2}{%  
2323 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%  
2324 }%  
2325 }
```

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

\GLSfirstplural

```
2326 \newrobustcmd*{\GLSfirstplural}{\@ifstar\@sGLSfirstplural\@GLSfirstplural}
```

Define the starred form:

```
2327 \newcommand*{\@sGLSfirstplural}[1][\@GLSfirstplural[hyper=false,#1]]
```

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

```
2328 \newcommand*{\@GLSfirstplural}[2][\@%  
2329 \new@ifnextchar[\@GLSfirstplural@{#1}{#2}]{\@GLSfirstplural@{#1}{#2}[]}]
```

Read in the final optional argument:

```
2330 \def\@GLSfirstplural@#1#2[#3]{%  
2331 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2332 \protected@edef\@glo@text{\glsentryfirstplural{#2}}%
```

Call \@gls@link

```
2333 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%  
2334 }%  
2335 }
```

\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

```
2336 \newrobustcmd*{\glsname}{\@ifstar\@sglsname\@glsname}
```

Define the starred form:

```
2337 \newcommand*{\@sglsname}[1][\@glsname[hyper=false,#1]]
```

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

```
2338 \newcommand*{\@glsname}[2][\@%  
2339 \new@ifnextchar[\@glsname@{#1}{#2}]{\@glsname@{#1}{#2}[]}]
```

Read in the final optional argument:

```
2340 \def\@glsname@#1#2[#3]{%  
2341 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2342 \protected@edef\@glo@text{\glsentryname{#2}}%
```

Call \@gls@link

```
2343 \@gls@link[#1]{#2}{\@glo@text#3}%  
2344 }%  
2345 }
```

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

\Glsname

```
2346 \newrobustcmd*{\Glsname}{\@ifstar\@sGlsname\@Glsname}
```

Define the starred form:

```
2347 \newcommand*{\@sGlsname}[1][\@Glsname[hyper=false,#1]]
```

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

```
2348 \newcommand*{\@Glsname}[2][\@%  
2349 \new@ifnextchar[\@Glsname@{#1}{#2}]{\@Glsname@{#1}{#2}[]}]
```

Read in the final optional argument:

```
2350 \def\@Glsname@#1#2[#3]{%  
2351 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2352 \protected@edef\@glo@text{\glsentryname{#2}}%
```

Call \@gls@link

```

2353 \@gls@link[#1]{#2}{%
2354   \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2355 }%
2356 }

```

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

\GLSname

```

2357 \newrobustcmd*{\GLSname}{\@ifstar\@sGLSname\@GLSname}

```

Define the starred form:

```

2358 \newcommand*{\@sGLSname}[1][\@GLSname[hyper=false,#1]]

```

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

```

2359 \newcommand*{\@GLSname}[2][\@GLSname[hyper=false,#1]]
2360 \new@ifnextchar[\@GLSname@{#1}{#2}]{\@GLSname@{#1}{#2}[]}

```

Read in the final optional argument:

```

2361 \def\@GLSname@#1#2[#3]{%
2362 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2363 \protected@edef\@glo@text{\glsentryname{#2}}%

```

Call \@gls@link

```

2364 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2365 }%
2366 }

```

\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

```

2367 \newrobustcmd*{\glsdesc}{\@ifstar\@sglsdesc\@glsdesc}

```

Define the starred form:

```

2368 \newcommand*{\@sglsdesc}[1][\@glsdesc[hyper=false,#1]]

```

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

```

2369 \newcommand*{\@glsdesc}[2][\@glsdesc[hyper=false,#1]]
2370 \new@ifnextchar[\@glsdesc@{#1}{#2}]{\@glsdesc@{#1}{#2}[]}

```

Read in the final optional argument:

```

2371 \def\@glsdesc@#1#2[#3]{%
2372 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2373 \protected@edef\@glo@text{\glsentrydesc{#2}}%

```

Call \@gls@link  
2374 \@gls@link[#1]{#2}{\@glo@text#3}%  
2375 }%  
2376 }

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

\Glsdesc

2377 \newrobustcmd\*{\Glsdesc}{\@ifstar\@sGlsdesc\@Glsdesc}

Define the starred form:

2378 \newcommand\*{\@sGlsdesc}[1][]{\@Glsdesc[hyper=false,#1]}

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

2379 \newcommand\*{\@Glsdesc}[2][]{%  
2380 \new@ifnextchar[{\@Glsdesc@{#1}{#2}}{\@Glsdesc@{#1}{#2}[]}]}

Read in the final optional argument:

2381 \def\@Glsdesc@#1#2[#3]{%  
2382 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

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

2383 \protected@edef\@glo@text{\glsentrydesc{#2}}%

Call \@gls@link

2384 \@gls@link[#1]{#2}{%  
2385 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%  
2386 }%  
2387 }

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

\GLSdesc

2388 \newrobustcmd\*{\GLSdesc}{\@ifstar\@sGLSdesc\@GLSdesc}

Define the starred form:

2389 \newcommand\*{\@sGLSdesc}[1][]{\@GLSdesc[hyper=false,#1]}

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

2390 \newcommand\*{\@GLSdesc}[2][]{%  
2391 \new@ifnextchar[{\@GLSdesc@{#1}{#2}}{\@GLSdesc@{#1}{#2}[]}]}

Read in the final optional argument:

2392 \def\@GLSdesc@#1#2[#3]{%  
2393 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

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

2394 \protected@edef\@glo@text{\glsentrydesc{#2}}%



Call \@gls@link

```
2395 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2396 }%
2397 }
```

\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

```
2398 \newrobustcmd*{\glsdescplural}{\@ifstar\sglsdescplural\glsdescplural}
```

Define the starred form:

```
2399 \newcommand*{\@sglsdescplural}[1][\@glsdescplural[hyper=false,#1]]
```

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

```
2400 \newcommand*{\@glsdescplural}[2][\@%
2401 \new@ifnextchar[\@glsdescplural@{#1}{#2}]{\@glsdescplural@{#1}{#2}[]}]
```

Read in the final optional argument:

```
2402 \def\@glsdescplural@#1#2[#3]{%
2403 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%
2404 \protected@edef\@glo@text{\glsentrydescplural{#2}}%
```

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

```
2404 \protected@edef\@glo@text{\glsentrydescplural{#2}}%
```

Call \@gls@link

```
2405 \@gls@link[#1]{#2}{\@glo@text#3}%
2406 }%
2407 }
```

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

\Glsdescplural

```
2408 \newrobustcmd*{\Glsdescplural}{\@ifstar\sglsdescplural\Glsdescplural}
```

Define the starred form:

```
2409 \newcommand*{\@sglsdescplural}[1][\@Glsdescplural[hyper=false,#1]]
```

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

```
2410 \newcommand*{\@Glsdescplural}[2][\@%
2411 \new@ifnextchar[\@Glsdescplural@{#1}{#2}]{\@Glsdescplural@{#1}{#2}[]}]
```

Read in the final optional argument:

```
2412 \def\@Glsdescplural@#1#2[#3]{%
2413 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%
2414 \protected@edef\@glo@text{\glsentrydescplural{#2}}%
```

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

```
2414 \protected@edef\@glo@text{\glsentrydescplural{#2}}%
```

Call \@gls@link

```

2415 \@gls@link[#1]{#2}{%
2416   \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2417 }%
2418 }

```

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

\GLSdescplural

```

2419 \newrobustcmd*{\GLSdescplural}{\@ifstar\@sGLSdescplural\@GLSdescplural}

```

Define the starred form:

```

2420 \newcommand*{\@sGLSdescplural}[1][\@GLSdescplural[hyper=false,#1]]

```

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

```

2421 \newcommand*{\@GLSdescplural}[2][\@GLSdescplural[hyper=false,#1]]
2422 \new@ifnextchar[\@GLSdescplural@{#1}{#2}]{\@GLSdescplural@{#1}{#2}[]}

```

Read in the final optional argument:

```

2423 \def\@GLSdescplural@#1#2[#3]{%
2424 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2425 \protected@edef\@glo@text{\glsentrydescplural{#2}}%

```

Call \@gls@link

```

2426 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2427 }%
2428 }

```

\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

```

2429 \newrobustcmd*{\glsymbol}{\@ifstar\@sglsymbol\@glsymbol}

```

Define the starred form:

```

2430 \newcommand*{\@sglsymbol}[1][\@glsymbol[hyper=false,#1]]

```

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

```

2431 \newcommand*{\@glsymbol}[2][\@glsymbol[hyper=false,#1]]
2432 \new@ifnextchar[\@glsymbol@{#1}{#2}]{\@glsymbol@{#1}{#2}[]}

```

Read in the final optional argument:

```

2433 \def\@glsymbol@#1#2[#3]{%
2434 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2435 \protected@edef\@glo@text{\glsentrysymbol{#2}}%

```

Call \@gls@link  
2436 \@gls@link[#1]{#2}{\@glo@text#3}%  
2437 }%  
2438 }

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

\Glsymbol

2439 \newrobustcmd\*{\Glsymbol}{\@ifstar\@sGlsymbol\@Glsymbol}

Define the starred form:

2440 \newcommand\*{\@sGlsymbol}[1][]{\@Glsymbol[hyper=false,#1]}

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

2441 \newcommand\*{\@Glsymbol}[2][]{%

2442 \new@ifnextchar[\@Glsymbol@{#1}{#2}]{\@Glsymbol@{#1}{#2}[]}}

Read in the final optional argument:

2443 \def\@Glsymbol@#1#2[#3]{%

2444 \glsoifexists{#2}{\edef\@glo@type{\glentrytype{#2}}%

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

2445 \protected@edef\@glo@text{\glentrysymbol{#2}}%

Call \@gls@link

2446 \@gls@link[#1]{#2}{%

2447 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%

2448 }%

2449 }

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

\GLSsymbol

2450 \newrobustcmd\*{\GLSsymbol}{\@ifstar\@sGLSsymbol\@GLSsymbol}

Define the starred form:

2451 \newcommand\*{\@sGLSsymbol}[1][]{\@GLSsymbol[hyper=false,#1]}

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

2452 \newcommand\*{\@GLSsymbol}[2][]{%

2453 \new@ifnextchar[\@GLSsymbol@{#1}{#2}]{\@GLSsymbol@{#1}{#2}[]}}

Read in the final optional argument:

2454 \def\@GLSsymbol@#1#2[#3]{%

2455 \glsoifexists{#2}{\edef\@glo@type{\glentrytype{#2}}%

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

2456 \protected@edef\@glo@text{\glentrysymbol{#2}}%

Call \@gls@link

```
2457 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%  
2458 }%  
2459 }
```

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

\glssymbolplural

```
2460 \newrobustcmd*{\glssymbolplural}{\@ifstar\sglssymbolplural\@glssymbolplural}
```

Define the starred form:

```
2461 \newcommand*{\@sglssymbolplural}[1] [] {\@glssymbolplural[hyper=false,#1]}
```

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

```
2462 \newcommand*{\@glssymbolplural}[2] [] {%  
2463 \new@ifnextchar [\@glssymbolplural@{#1}{#2}]{\@glssymbolplural@{#1}{#2} []}}
```

Read in the final optional argument:

```
2464 \def\@glssymbolplural@#1#2[#3] {%  
2465 \glsdoifexists{#2}{\edef\@glo@type{\glstrytype{#2}}%
```

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

```
2466 \protected@edef\@glo@text{\glstrysymbolplural{#2}}%
```

Call \@gls@link

```
2467 \@gls@link[#1]{#2}{\@glo@text#3}%  
2468 }%  
2469 }
```

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

\Glsymbolplural

```
2470 \newrobustcmd*{\Glsymbolplural}{\@ifstar\sglsymbolplural\@Glsymbolplural}
```

Define the starred form:

```
2471 \newcommand*{\sglsymbolplural}[1] [] {\@Glsymbolplural[hyper=false,#1]}
```

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

```
2472 \newcommand*{\@Glsymbolplural}[2] [] {%  
2473 \new@ifnextchar [\@Glsymbolplural@{#1}{#2}]{\@Glsymbolplural@{#1}{#2} []}}
```

Read in the final optional argument:

```
2474 \def\@Glsymbolplural@#1#2[#3] {%  
2475 \glsdoifexists{#2}{\edef\@glo@type{\glstrytype{#2}}%
```

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

```
2476 \protected@edef\@glo@text{\glstrysymbolplural{#2}}%
```

Call \@gls@link

```

2477 \@gls@link[#1]{#2}{%
2478   \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2479 }%
2480 }

```

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

\GLSsymbolplural

```

2481 \newrobustcmd*{\GLSsymbolplural}{\@ifstar\@sGLSsymbolplural\@GLSsymbolplural}

```

Define the starred form:

```

2482 \newcommand*{\@sGLSsymbolplural}[1][\@GLSsymbolplural[hyper=false,#1]]{

```

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

```

2483 \newcommand*{\@GLSsymbolplural}[2][\@GLSsymbolplural@{#1}{#2}[]]{%
2484 \new@ifnextchar[\@GLSsymbolplural@{#1}{#2}]{\@GLSsymbolplural@{#1}{#2}[]]}

```

Read in the final optional argument:

```

2485 \def\@GLSsymbolplural@#1#2[#3]{%
2486 \glsdoifexists{#2}{\edef\@glo@type{\glstrytype{#2}}}%

```

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

```

2487 \protected@edef\@glo@text{\glstrysymbolplural{#2}}%

```

Call \@gls@link

```

2488 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2489 }%
2490 }

```

\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

```

2491 \newrobustcmd*{\glsuseri}{\@ifstar\@sglsuseri\@glsuseri}

```

Define the starred form:

```

2492 \newcommand*{\@sglsuseri}[1][\@glsuseri[hyper=false,#1]]{

```

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

```

2493 \newcommand*{\@glsuseri}[2][\@glsuseri@{#1}{#2}[]]{%
2494 \new@ifnextchar[\@glsuseri@{#1}{#2}]{\@glsuseri@{#1}{#2}[]]}

```

Read in the final optional argument:

```

2495 \def\@glsuseri@#1#2[#3]{%
2496 \glsdoifexists{#2}{\edef\@glo@type{\glstrytype{#2}}}%

```

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

```

2497 \protected@edef\@glo@text{\glstryuseri{#2}}%

```

Call \@gls@link  
2498 \@gls@link[#1]{#2}{\@glo@text#3}%  
2499 }%  
2500 }

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

\Glsuseri

2501 \newrobustcmd\*{\Glsuseri}{\@ifstar\@sGlsuseri\@Glsuseri}

Define the starred form:

2502 \newcommand\*{\@sGlsuseri}[1][ ]{\@Glsuseri[hyper=false,#1]}

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

2503 \newcommand\*{\@Glsuseri}[2][ ]{%

2504 \new@ifnextchar[{\@Glsuseri@{#1}{#2}}{\@Glsuseri@{#1}{#2}[ ]}]

Read in the final optional argument:

2505 \def\@Glsuseri@#1#2[#3]{%

2506 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

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

2507 \protected@edef\@glo@text{\glsentryuseri{#2}}%

Call \@gls@link

2508 \@gls@link[#1]{#2}{%

2509 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%

2510 }%

2511 }

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

\GLSuseri

2512 \newrobustcmd\*{\GLSuseri}{\@ifstar\@sGLSuseri\@GLSuseri}

Define the starred form:

2513 \newcommand\*{\@sGLSuseri}[1][ ]{\@GLSuseri[hyper=false,#1]}

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

2514 \newcommand\*{\@GLSuseri}[2][ ]{%

2515 \new@ifnextchar[{\@GLSuseri@{#1}{#2}}{\@GLSuseri@{#1}{#2}[ ]}]

Read in the final optional argument:

2516 \def\@GLSuseri@#1#2[#3]{%

2517 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

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

2518 \protected@edef\@glo@text{\glsentryuseri{#2}}%

Call \@gls@link

```

2519 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2520 }%
2521 }

```

\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

```

2522 \newrobustcmd*{\glsuserii}{\@ifstar\@sglsuserii\@glsuserii}

```

Define the starred form:

```

2523 \newcommand*{\@sglsuserii}[1][\@glsuserii[hyper=false,#1]]{

```

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

```

2524 \newcommand*{\@glsuserii}[2][\@%
2525 \new@ifnextchar[\@glsuserii@{#1}{#2}]{\@glsuserii@{#1}{#2}[]}]

```

Read in the final optional argument:

```

2526 \def\@glsuserii@#1#2[#3]{%
2527 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2528 \protected@edef\@glo@text{\glsentryuserii{#2}}%

```

Call \@gls@link

```

2529 \@gls@link[#1]{#2}{\@glo@text#3}%
2530 }%
2531 }

```

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

\Glsuserii

```

2532 \newrobustcmd*{\Glsuserii}{\@ifstar\@sGlsuserii\@Glsuserii}

```

Define the starred form:

```

2533 \newcommand*{\@sGlsuserii}[1][\@Glsuserii[hyper=false,#1]]{

```

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

```

2534 \newcommand*{\@Glsuserii}[2][\@%
2535 \new@ifnextchar[\@Glsuserii@{#1}{#2}]{\@Glsuserii@{#1}{#2}[]}]

```

Read in the final optional argument:

```

2536 \def\@Glsuserii@#1#2[#3]{%
2537 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2538 \protected@edef\@glo@text{\glsentryuserii{#2}}%

```

Call \@gls@link

```

2539 \@gls@link[#1]{#2}{%
2540   \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2541 }%
2542 }

```

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

\GLSuserii

```

2543 \newrobustcmd*{\GLSuserii}{\@ifstar\@sGLSuserii\@GLSuserii}

```

Define the starred form:

```

2544 \newcommand*{\@sGLSuserii}[1][\@GLSuserii[hyper=false,#1]]{

```

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

```

2545 \newcommand*{\@GLSuserii}[2][\@GLSuserii@{#1}{#2}[]]{%
2546 \new@ifnextchar[\@GLSuserii@{#1}{#2}}{\@GLSuserii@{#1}{#2}[]]}

```

Read in the final optional argument:

```

2547 \def\@GLSuserii@#1#2[#3]{%
2548 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2549 \protected@edef\@glo@text{\glsentryuserii{#2}}%

```

Call \@gls@link

```

2550 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2551 }%
2552 }

```

\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

```

2553 \newrobustcmd*{\glsuseriii}{\@ifstar\@sglsuseriii\@glsuseriii}

```

Define the starred form:

```

2554 \newcommand*{\@sglsuseriii}[1][\@glsuseriii[hyper=false,#1]]{

```

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

```

2555 \newcommand*{\@glsuseriii}[2][\@glsuseriii@{#1}{#2}[]]{%
2556 \new@ifnextchar[\@glsuseriii@{#1}{#2}}{\@glsuseriii@{#1}{#2}[]]}

```

Read in the final optional argument:

```

2557 \def\@glsuseriii@#1#2[#3]{%
2558 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2559 \protected@edef\@glo@text{\glsentryuseriii{#2}}%

```



Call \@gls@link  
2560 \@gls@link[#1]{#2}{\@glo@text#3}%  
2561 }%  
2562 }

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

\Glsuseriii

2563 \newrobustcmd\*{\Glsuseriii}{\@ifstar\@sGlsuseriii\@Glsuseriii}

Define the starred form:

2564 \newcommand\*{\@sGlsuseriii}[1][ ]{\@Glsuseriii[hyper=false,#1]}

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

2565 \newcommand\*{\@Glsuseriii}[2][ ]{%  
2566 \new@ifnextchar[{\@Glsuseriii@{#1}{#2}}{\@Glsuseriii@{#1}{#2}[ ]}]}

Read in the final optional argument:

2567 \def\@Glsuseriii@#1#2[#3]{%  
2568 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

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

2569 \protected@edef\@glo@text{\glsentryuseriii{#2}}%

Call \@gls@link

2570 \@gls@link[#1]{#2}{%  
2571 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%  
2572 }%  
2573 }

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

\GLSuseriii

2574 \newrobustcmd\*{\GLSuseriii}{\@ifstar\@sGLSuseriii\@GLSuseriii}

Define the starred form:

2575 \newcommand\*{\@sGLSuseriii}[1][ ]{\@GLSuseriii[hyper=false,#1]}

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

2576 \newcommand\*{\@GLSuseriii}[2][ ]{%  
2577 \new@ifnextchar[{\@GLSuseriii@{#1}{#2}}{\@GLSuseriii@{#1}{#2}[ ]}]}

Read in the final optional argument:

2578 \def\@GLSuseriii@#1#2[#3]{%  
2579 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

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

2580 \protected@edef\@glo@text{\glsentryuseriii{#2}}%

Call \@gls@link

```
2581 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%  
2582 }%  
2583 }
```

\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

```
2584 \newrobustcmd*{\glsuseriv}{\@ifstar\@sglsuseriv\@glsuseriv}
```

Define the starred form:

```
2585 \newcommand*{\@sglsuseriv}[1][\@glsuseriv[hyper=false,#1]]
```

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

```
2586 \newcommand*{\@glsuseriv}[2][\@%  
2587 \new@ifnextchar[\@glsuseriv@{#1}{#2}]{\@glsuseriv@{#1}{#2}[]}]
```

Read in the final optional argument:

```
2588 \def\@glsuseriv@#1#2[#3]{%  
2589 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2590 \protected@edef\@glo@text{\glsentryuseriv{#2}}%
```

Call \@gls@link

```
2591 \@gls@link[#1]{#2}{\@glo@text#3}%  
2592 }%  
2593 }
```

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

\Glsuseriv

```
2594 \newrobustcmd*{\Glsuseriv}{\@ifstar\@sGlsuseriv\@Glsuseriv}
```

Define the starred form:

```
2595 \newcommand*{\@sGlsuseriv}[1][\@Glsuseriv[hyper=false,#1]]
```

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

```
2596 \newcommand*{\@Glsuseriv}[2][\@%  
2597 \new@ifnextchar[\@Glsuseriv@{#1}{#2}]{\@Glsuseriv@{#1}{#2}[]}]
```

Read in the final optional argument:

```
2598 \def\@Glsuseriv@#1#2[#3]{%  
2599 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2600 \protected@edef\@glo@text{\glsentryuseriv{#2}}%
```

Call \@gls@link

```

2601 \@gls@link[#1]{#2}{%
2602   \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2603 }%
2604 }

```

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

\GLSuseriv

```

2605 \newrobustcmd*{\GLSuseriv}{\@ifstar\@sGLSuseriv\@GLSuseriv}

```

Define the starred form:

```

2606 \newcommand*{\@sGLSuseriv}[1][\@GLSuseriv[hyper=false,#1]]{

```

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

```

2607 \newcommand*{\@GLSuseriv}[2][\@GLSuseriv@{#1}{#2}[]]{%
2608 \new@ifnextchar[\@GLSuseriv@{#1}{#2}]{\@GLSuseriv@{#1}{#2}[]}}

```

Read in the final optional argument:

```

2609 \def\@GLSuseriv@#1#2[#3]{%
2610 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2611 \protected@edef\@glo@text{\glsentryuseriv{#2}}%

```

Call \@gls@link

```

2612 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2613 }%
2614 }

```

\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

```

2615 \newrobustcmd*{\glsuserv}{\@ifstar\@sglsuserv\@glsuserv}

```

Define the starred form:

```

2616 \newcommand*{\@sglsuserv}[1][\@glsuserv[hyper=false,#1]]{

```

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

```

2617 \newcommand*{\@glsuserv}[2][\@glsuserv@{#1}{#2}[]]{%
2618 \new@ifnextchar[\@glsuserv@{#1}{#2}]{\@glsuserv@{#1}{#2}[]}}

```

Read in the final optional argument:

```

2619 \def\@glsuserv@#1#2[#3]{%
2620 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2621 \protected@edef\@glo@text{\glsentryuseriv{#2}}%

```

Call \@gls@link  
2622 \@gls@link[#1]{#2}{\@glo@text#3}%  
2623 }%  
2624 }

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

\Glsuserv

2625 \newrobustcmd\*{\Glsuserv}{\@ifstar\@sGlsuserv\@Glsuserv}

Define the starred form:

2626 \newcommand\*{\@sGlsuserv}[1][ ]{\@Glsuserv[hyper=false,#1]}

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

2627 \newcommand\*{\@Glsuserv}[2][ ]{%  
2628 \new@ifnextchar[{\@Glsuserv@{#1}{#2}}{\@Glsuserv@{#1}{#2}[ ]}]}

Read in the final optional argument:

2629 \def\@Glsuserv@#1#2[#3]{%  
2630 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

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

2631 \protected@edef\@glo@text{\glsentryuser{#2}}%

Call \@gls@link

2632 \@gls@link[#1]{#2}{%  
2633 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%  
2634 }%  
2635 }

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

\GLSuserv

2636 \newrobustcmd\*{\GLSuserv}{\@ifstar\@sGLSuserv\@GLSuserv}

Define the starred form:

2637 \newcommand\*{\@sGLSuserv}[1][ ]{\@GLSuserv[hyper=false,#1]}

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

2638 \newcommand\*{\@GLSuserv}[2][ ]{%  
2639 \new@ifnextchar[{\@GLSuserv@{#1}{#2}}{\@GLSuserv@{#1}{#2}[ ]}]}

Read in the final optional argument:

2640 \def\@GLSuserv@#1#2[#3]{%  
2641 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%  
2642 \protected@edef\@glo@text{\glsentryuser{#2}}%

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

2642 \protected@edef\@glo@text{\glsentryuser{#2}}%

Call \@gls@link

```
2643 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%  
2644 }%  
2645 }
```

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

\glsuservi

```
2646 \newrobustcmd*{\glsuservi}{\@ifstar\@sglsuservi\@glsuservi}
```

Define the starred form:

```
2647 \newcommand*{\@sglsuservi}[1][]{\@glsuservi[hyper=false,#1]}
```

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

```
2648 \newcommand*{\@glsuservi}[2][]{%  
2649 \new@ifnextchar[{\@glsuservi@{#1}{#2}}{\@glsuservi@{#1}{#2}[]}]}
```

Read in the final optional argument:

```
2650 \def\@glsuservi@#1#2[#3]{%  
2651 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2652 \protected@edef\@glo@text{\glsentryuservi{#2}}%
```

Call \@gls@link

```
2653 \@gls@link[#1]{#2}{\@glo@text#3}%  
2654 }%  
2655 }
```

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

\Glsuservi

```
2656 \newrobustcmd*{\Glsuservi}{\@ifstar\@sGlsuservi\@Glsuservi}
```

Define the starred form:

```
2657 \newcommand*{\@sGlsuservi}[1][]{\@Glsuservi[hyper=false,#1]}
```

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

```
2658 \newcommand*{\@Glsuservi}[2][]{%  
2659 \new@ifnextchar[{\@Glsuservi@{#1}{#2}}{\@Glsuservi@{#1}{#2}[]}]}
```

Read in the final optional argument:

```
2660 \def\@Glsuservi@#1#2[#3]{%  
2661 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2662 \protected@edef\@glo@text{\glsentryuservi{#2}}%
```

Call \@gls@link

```

2663 \@gls@link[#1]{#2}{%
2664   \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2665 }%
2666 }

```

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

\GLSuservi

```

2667 \newrobustcmd*{\GLSuservi}{\@ifstar\@sGLSuservi\@GLSuservi}

```

Define the starred form:

```

2668 \newcommand*{\@sGLSuservi}[1] [] {\@GLSuservi[hyper=false,#1]}

```

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

```

2669 \newcommand*{\@GLSuservi}[2] [] {%
2670 \new@ifnextchar [\@GLSuservi@{#1}{#2}]{\@GLSuservi@{#1}{#2} []}}

```

Read in the final optional argument:

```

2671 \def\@GLSuservi@#1#2[#3] {%
2672 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2673 \protected@edef\@glo@text{\glsentryuservi{#2}}%

```

Call \@gls@link

```

2674 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2675 }%
2676 }

```

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

\acrshort

```

2677 \newrobustcmd*{\acrshort}{\@ifstar\s@acrshort\@ns@acrshort}

```

Define the starred form:

```

2678 \newcommand*{\s@acrshort}[2] [] {%
2679   \new@ifnextchar [\@acrshort{hyper=false,#1}{#2}}%
2680   {\@acrshort{hyper=false,#1}{#2} []}%
2681 }

```

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

```

2682 \newcommand*{\@ns@acrshort}[2] [] {%
2683   \new@ifnextchar [\@acrshort{#1}{#2}]{\@acrshort{#1}{#2} []}%
2684 }

```

Read in the final optional argument:

```

2685 \def\@acrshort#1#2[#3] {%
2686   \glsdoifexists{#2}%
2687   {%
2688     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```
2689 \protected@edef\@glo@text{\glentryshort{#2}}%  
    Call \@gls@link  
2690 \@gls@link[#1]{#2}{\acronymfont{\@glo@text}#3}%  
2691 }%  
2692 }
```

\Acrshort

```
2693 \newrobustcmd*{\Acrshort}{\@ifstar\s@Acrshort\ns@Acrshort}
```

Define the starred form:

```
2694 \newcommand*{\s@Acrshort}[2][]{%  
2695 \new@ifnextchar[{\@Acrshort{hyper=false,#1}{#2}}%  
2696 {\@Acrshort{hyper=false,#1}{#2}[]}%  
2697 }
```

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

```
2698 \newcommand*{\ns@Acrshort}[2][]{%  
2699 \new@ifnextchar[{\@Acrshort{#1}{#2}}{\@Acrshort{#1}{#2}[]}%  
2700 }
```

Read in the final optional argument:

```
2701 \def\@Acrshort#1#2[#3]{%  
2702 \glsoifexists{#2}%  
2703 {%  
2704 \edef\@glo@type{\glentrytype{#2}}%
```

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

```
2705 \protected@edef\@glo@text{\glentryshort{#2}}%  
    Call \@gls@link  
2706 \@gls@link[#1]{#2}%  
2707 {%  
2708 \acronymfont{\expandafter\makefirstuc\expandafter{\@glo@text}}#3%  
2709 }%  
2710 }%  
2711 }
```

\ACRshort

```
2712 \newrobustcmd*{\ACRshort}{\@ifstar\s@ACRshort\ns@ACRshort}
```

Define the starred form:

```
2713 \newcommand*{\s@ACRshort}[2][]{%  
2714 \new@ifnextchar[{\@ACRshort{hyper=false,#1}{#2}}%  
2715 {\@ACRshort{hyper=false,#1}{#2}[]}%  
2716 }
```

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

```
2717 \newcommand*{\ns@ACRshort}[2] [] {%
2718   \new@ifnextchar[{\@ACRshort{#1}{#2}}{\@ACRshort{#1}{#2} []}%
2719 }
```

Read in the final optional argument:

```
2720 \def\@ACRshort#1#2[#3] {%
2721   \glsdoifexists{#2}%
2722   {%
2723     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2724   \protected@edef\@glo@text{\glsentryshort{#2}}%
```

Call \@gls@link

```
2725   \@gls@link[#1]{#2}{\acronymfont{\MakeUppercase{\@glo@text#3}}}%
2726   }%
2727 }
```

Short plural:

\acrshortpl

```
2728 \newrobustcmd*{\acrshortpl}{\@ifstar\s@acrshortpl\ns@acrshortpl}
```

Define the starred form:

```
2729 \newcommand*{\s@acrshortpl}[2] [] {%
2730   \new@ifnextchar[{\@acrshortpl{hyper=false,#1}{#2}}%
2731   {\@acrshortpl{hyper=false,#1}{#2} []}%
2732 }
```

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

```
2733 \newcommand*{\ns@acrshortpl}[2] [] {%
2734   \new@ifnextchar[{\@acrshortpl{#1}{#2}}{\@acrshortpl{#1}{#2} []}%
2735 }
```

Read in the final optional argument:

```
2736 \def\@acrshortpl#1#2[#3] {%
2737   \glsdoifexists{#2}%
2738   {%
2739     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2740   \protected@edef\@glo@text{\glsentryshortpl{#2}}%
```

Call \@gls@link

```
2741   \@gls@link[#1]{#2}{\acronymfont{\@glo@text#3}}%
2742   }%
2743 }
```

\Acrshortpl

```
2744 \newrobustcmd*{\Acrshortpl}{\@ifstar\s@Acrshortpl\ns@Acrshortpl}
```



Define the starred form:

```
2745 \newcommand*{\s@Acrshortpl}[2] [] {%
2746   \new@ifnextchar[{\@Acrshortpl{hyper=false,#1}{#2}}%
2747     {\@Acrshortpl{hyper=false,#1}{#2} []}%
2748 }
```

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

```
2749 \newcommand*{\ns@Acrshortpl}[2] [] {%
2750   \new@ifnextchar[{\@Acrshortpl{#1}{#2}}{\@Acrshortpl{#1}{#2} []}%
2751 }
```

Read in the final optional argument:

```
2752 \def\@Acrshortpl#1#2[#3] {%
2753   \glsdoifexists{#2}%
2754   {%
2755     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2756   \protected@edef\@glo@text{\glsentryshortpl{#2}}%
```

Call \@gls@link

```
2757   \@gls@link[#1]{#2}%
2758   {%
2759     \acronymfont{\expandafter\makefirstuc\expandafter{\@glo@text}}#3%
2760   }%
2761 }%
2762 }
```

\ACRshortpl

```
2763 \newrobustcmd*{\ACRshortpl}{\@ifstar\s@ACRshortpl\ns@ACRshortpl}
```

Define the starred form:

```
2764 \newcommand*{\s@ACRshortpl}[2] [] {%
2765   \new@ifnextchar[{\@ACRshortpl{hyper=false,#1}{#2}}%
2766     {\@ACRshortpl{hyper=false,#1}{#2} []}%
2767 }
```

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

```
2768 \newcommand*{\ns@ACRshortpl}[2] [] {%
2769   \new@ifnextchar[{\@ACRshortpl{#1}{#2}}{\@ACRshortpl{#1}{#2} []}%
2770 }
```

Read in the final optional argument:

```
2771 \def\@ACRshortpl#1#2[#3] {%
2772   \glsdoifexists{#2}%
2773   {%
2774     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2775   \protected@edef\@glo@text{\glsentryshortpl{#2}}%
```

Call \@gls@link

```
2776   \@gls@link[#1]{#2}{\acronymfont{\MakeUppercase{\@glo@text#3}}}%  
2777   }%  
2778 }
```

\acrlong

```
2779 \newrobustcmd*{\acrlong}{\@ifstar\s@acrlong\ns@acrlong}
```

Define the starred form:

```
2780 \newcommand*{\s@acrlong}[2][]{%  
2781   \new@ifnextchar[{\@acrlong{hyper=false,#1}{#2}}%  
2782     {\@acrlong{hyper=false,#1}{#2}[]}%  
2783 }
```

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

```
2784 \newcommand*{\ns@acrlong}[2][]{%  
2785   \new@ifnextchar[{\@acrlong{#1}{#2}}{\@acrlong{#1}{#2}[]}%  
2786 }
```

Read in the final optional argument:

```
2787 \def\@acrlong#1#2[#3]{%  
2788   \glsdoifexists{#2}%  
2789   {%  
2790     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2791   \protected@edef\@glo@text{\glsentrylong{#2}}%
```

Call \@gls@link

```
2792   \@gls@link[#1]{#2}{\@glo@text#3}%  
2793   }%  
2794 }
```

\Acrlong

```
2795 \newrobustcmd*{\Acrlong}{\@ifstar\s@Acrlong\ns@Acrlong}
```

Define the starred form:

```
2796 \newcommand*{\s@Acrlong}[2][]{%  
2797   \new@ifnextchar[{\@Acrlong{hyper=false,#1}{#2}}%  
2798     {\@Acrlong{hyper=false,#1}{#2}[]}%  
2799 }
```

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

```
2800 \newcommand*{\ns@Acrlong}[2][]{%  
2801   \new@ifnextchar[{\@Acrlong{#1}{#2}}{\@Acrlong{#1}{#2}[]}%  
2802 }
```

Read in the final optional argument:

```
2803 \def\@Acrlong#1#2[#3]{%
2804   \glsdoifexists{#2}%
2805   {%
2806     \edef\@glo@type{\glstrytype{#2}}%
2807     \protected@edef\@glo@text{\glstrylong{#2}}%
2808     \Call\@gls@link
2809     {%
2810       \expandafter\makefirstuc\expandafter{\@glo@text}#3%
2811     }%
2812   }%
2813 }
```

\ACRlong

```
2814 \newrobustcmd*{\ACRlong}{\@ifstar\s@ACRlong\ns@ACRlong}
```

Define the starred form:

```
2815 \newcommand*{\s@ACRlong}[2][{}]{%
2816   \new@ifnextchar[{\@ACRlong{hyper=false,#1}{#2}}%
2817   {\@ACRlong{hyper=false,#1}{#2}[]}%
2818 }
```

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

```
2819 \newcommand*{\ns@ACRlong}[2][{}]{%
2820   \new@ifnextchar[{\@ACRlong{#1}{#2}}{\@ACRlong{#1}{#2}[]}%
2821 }
```

Read in the final optional argument:

```
2822 \def\@ACRlong#1#2[#3]{%
2823   \glsdoifexists{#2}%
2824   {%
2825     \edef\@glo@type{\glstrytype{#2}}%
2826     \protected@edef\@glo@text{\glstrylong{#2}}%
2827     \Call\@gls@link
2828     {%
2829       \@gls@link{#1}{#2}{\MakeUppercase{\@glo@text#3}}%
2830     }%
2831   }
```

Short plural:

\acrlongpl

```
2830 \newrobustcmd*{\acrlongpl}{\@ifstar\s@acrlongpl\ns@acrlongpl}
```

Define the starred form:

```
2831 \newcommand*{\s@acrlongpl}[2] [] {%
2832   \new@ifnextchar[{\@acrlongpl{hyper=false,#1}{#2}}}%
2833   {\@acrlongpl{hyper=false,#1}{#2} []}%
2834 }
```

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

```
2835 \newcommand*{\ns@acrlongpl}[2] [] {%
2836   \new@ifnextchar[{\@acrlongpl{#1}{#2}}{\@acrlongpl{#1}{#2} []}%
2837 }
```

Read in the final optional argument:

```
2838 \def\@acrlongpl#1#2[#3] {%
2839   \glsdoifexists{#2}%
2840   {%
2841     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2842   \protected@edef\@glo@text{\glsentrylongpl{#2}}%
```

Call \@gls@link

```
2843   \@gls@link[#1]{#2}{\@glo@text#3}%
2844   }%
2845 }
```

\Acrlongpl

```
2846 \newrobustcmd*{\Acrlongpl}{\@ifstar\s@Acrlongpl\ns@Acrlongpl}
```

Define the starred form:

```
2847 \newcommand*{\s@Acrlongpl}[2] [] {%
2848   \new@ifnextchar[{\@Acrlongpl{hyper=false#1}{#2}}}%
2849   {\@Acrlongpl{hyper=false,#1}{#2} []}%
2850 }
```

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

```
2851 \newcommand*{\ns@Acrlongpl}[2] [] {%
2852   \new@ifnextchar[{\@Acrlongpl{#1}{#2}}{\@Acrlongpl{#1}{#2} []}%
2853 }
```

Read in the final optional argument:

```
2854 \def\@Acrlongpl#1#2[#3] {%
2855   \glsdoifexists{#2}%
2856   {%
2857     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2858   \protected@edef\@glo@text{\glsentrylongpl{#2}}%
```

```

Call \@gls@link
2859   \@gls@link[#1]{#2}%
2860   {%
2861     \expandafter\makefirstuc\expandafter{\@glo@text}#3%
2862   }%
2863 }%
2864 }

```

`\ACRlongpl`

```
2865 \newrobustcmd*{\ACRlongpl}{\@ifstar\s@ACRlongpl\ns@ACRlongpl}
```

Define the starred form:

```

2866 \newcommand*{\s@ACRlongpl}[2][ ]{%
2867   \new@ifnextchar[{\@ACRlongpl{hyper=false,#1}{#2}}%
2868     {\@ACRlongpl{hyper=false,#1}{#2}[ ]}%
2869 }

```

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

```

2870 \newcommand*{\ns@ACRlongpl}[2][ ]{%
2871   \new@ifnextchar[{\@ACRlongpl{#1}{#2}}{\@ACRlongpl{#1}{#2}[ ]}%
2872 }

```

Read in the final optional argument:

```

2873 \def\@ACRlongpl#1#2[#3]{%
2874   \glsdoifexists{#2}%
2875   {%
2876     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```
2877   \protected@edef\@glo@text{\glsentrylongpl{#2}}%
```

Call `\@gls@link`

```

2878   \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2879 }%
2880 }

```

### 1.10.2 Displaying entry details without adding information to the glossary

These commands merely display entry information without adding entries in the associated file or having hyperlinks.

Get the entry name (as specified by the name key when the entry was defined). The argument is the label associated with the entry. Note that unless you used `name=false` in the `sanitize` package option you may get unexpected results if the name key contains any commands.

`\glsentryname`

```
2881 \newcommand*{\glsentryname}[1]{\csname glo@#1@name\endcsname}
```

`\Glsentryname`

```
2882 \newcommand*{\Glsentryname}[1]{%
2883 \protected@edef\@glo@text{\csname glo@#1@name\endcsname}%
2884 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

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`

```
2885 \newcommand*{\glsentrydesc}[1]{\csname glo@#1@desc\endcsname}
```

`\Glsentrydesc`

```
2886 \newcommand*{\Glsentrydesc}[1]{%
2887 \protected@edef\@glo@text{\csname glo@#1@desc\endcsname}%
2888 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Plural form:

`\glsentrydescplural`

```
2889 \newcommand*{\glsentrydescplural}[1]{%
2890 \csname glo@#1@descplural\endcsname}
```

`\Glsentrydescplural`

```
2891 \newcommand*{\Glsentrydescplural}[1]{%
2892 \protected@edef\@glo@text{\csname glo@#1@descplural\endcsname}%
2893 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Get the entry text, as specified by the text key when the entry was defined. The argument is the label associated with the entry:

`\glsentrytext`

```
2894 \newcommand*{\glsentrytext}[1]{\csname glo@#1@text\endcsname}
```

`\Glsentrytext`

```
2895 \newcommand*{\Glsentrytext}[1]{%
2896 \protected@edef\@glo@text{\csname glo@#1@text\endcsname}%
2897 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Get the plural form:

`\glsentryplural`

```
2898 \newcommand*{\glsentryplural}[1]{\csname glo@#1@plural\endcsname}
```

`\Glsentryplural`

```
2899 \newcommand*{\Glsentryplural}[1]{%
2900 \protected@edef\@glo@text{\csname glo@#1@plural\endcsname}%
2901 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Get the symbol associated with this entry. The argument is the label associated with the entry. Note that unless you used `symbol=false` in the `sanitize` package option you may get unexpected results if the symbol key contained any commands.

`\glsentrysymbol`

```
2902 \newcommand*{\glsentrysymbol}[1]{\csname glo@#1@symbol\endcsname}
```

`\Glsentrysymbol`

```
2903 \newcommand*{\Glsentrysymbol}[1]{%
2904 \protected@edef\@glo@text{\csname glo@#1@symbol\endcsname}%
2905 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Plural form:

`\glsentrysymbolplural`

```
2906 \newcommand*{\glsentrysymbolplural}[1]{%
2907 \csname glo@#1@symbolplural\endcsname}
```

`\Glsentrysymbolplural`

```
2908 \newcommand*{\Glsentrysymbolplural}[1]{%
2909 \protected@edef\@glo@text{\csname glo@#1@symbolplural\endcsname}%
2910 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Get the entry text to be used when the entry is first used in the document (as specified by the first key when the entry was defined).

`\glsentryfirst`

```
2911 \newcommand*{\glsentryfirst}[1]{\csname glo@#1@first\endcsname}
```

`\Glsentryfirst`

```
2912 \newcommand*{\Glsentryfirst}[1]{%
2913 \protected@edef\@glo@text{\csname glo@#1@first\endcsname}%
2914 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

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

`\glsentryfirstplural`

```
2915 \newcommand*{\glsentryfirstplural}[1]{%
2916 \csname glo@#1@firstpl\endcsname}
```

`\Glsentryfirstplural`

```
2917 \newcommand*{\Glsentryfirstplural}[1]{%
2918 \protected@edef\@glo@text{\csname glo@#1@firstpl\endcsname}%
2919 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

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

`\glentrytype`

2920 `\newcommand*{\glentrytype}[1]{\csname glo@#1@type\endcsname}`

Display the sort text used for this entry. Note that the sort key is sanitize, so unexpected results may occur if the sort key contained commands.

`\glentrysort`

2921 `\newcommand*{\glentrysort}[1]{\csname glo@#1@sort\endcsname}`

`\glentryuseri` Get the first user key (as specified by the user1 when the entry was defined). The argument is the label associated with the entry.

2922 `\newcommand*{\glentryuseri}[1]{\csname glo@#1@useri\endcsname}`

`\Glsentryuseri`

2923 `\newcommand*{\Glsentryuseri}[1]{%`  
2924 `\protected@edef\@glo@text{\csname glo@#1@useri\endcsname}%`  
2925 `\expandafter\makefirstuc\expandafter{\@glo@text}}`

`\glentryuserii` Get the second user key (as specified by the user2 when the entry was defined). The argument is the label associated with the entry.

2926 `\newcommand*{\glentryuserii}[1]{\csname glo@#1@userii\endcsname}`

`\Glsentryuserii`

2927 `\newcommand*{\Glsentryuserii}[1]{%`  
2928 `\protected@edef\@glo@text{\csname glo@#1@userii\endcsname}%`  
2929 `\expandafter\makefirstuc\expandafter{\@glo@text}}`

`\glentryuseriii` Get the third user key (as specified by the user3 when the entry was defined). The argument is the label associated with the entry.

2930 `\newcommand*{\glentryuseriii}[1]{\csname glo@#1@useriii\endcsname}`

`\Glsentryuseriii`

2931 `\newcommand*{\Glsentryuseriii}[1]{%`  
2932 `\protected@edef\@glo@text{\csname glo@#1@useriii\endcsname}%`  
2933 `\expandafter\makefirstuc\expandafter{\@glo@text}}`

`\glentryuseriv` Get the fourth user key (as specified by the user4 when the entry was defined). The argument is the label associated with the entry.

2934 `\newcommand*{\glentryuseriv}[1]{\csname glo@#1@useriv\endcsname}`

`\Glsentryuseriv`

2935 `\newcommand*{\Glsentryuseriv}[1]{%`  
2936 `\protected@edef\@glo@text{\csname glo@#1@useriv\endcsname}%`  
2937 `\expandafter\makefirstuc\expandafter{\@glo@text}}`

`\glentryuserv` Get the fifth user key (as specified by the user5 when the entry was defined). The argument is the label associated with the entry.

2938 `\newcommand*{\glentryuserv}[1]{\csname glo@#1@userv\endcsname}`



```

\Glsentryuserv
2939 \newcommand*{\Glsentryuserv}[1]{%
2940 \protected@edef\@glo@text{\csname glo@#1@userv\endcsname}%
2941 \expandafter\makefirstuc\expandafter{\@glo@text}}

\glsentryuservi  Get the sixth user key (as specified by the user6 when the entry was defined).
                  The argument is the label associated with the entry.
2942 \newcommand*{\glsentryuservi}[1]{\csname glo@#1@uservi\endcsname}

\Glsentryuservi
2943 \newcommand*{\Glsentryuservi}[1]{%
2944 \protected@edef\@glo@text{\csname glo@#1@uservi\endcsname}%
2945 \expandafter\makefirstuc\expandafter{\@glo@text}}

\glsentryshort  Get the short key (as specified by the short the entry was defined). The argu-
                  ment is the label associated with the entry.
2946 \newcommand*{\glsentryshort}[1]{\csname glo@#1@short\endcsname}

\Glsentryshort
2947 \newcommand*{\Glsentryshort}[1]{%
2948 \protected@edef\@glo@text{\csname glo@#1@short\endcsname}%
2949 \expandafter\makefirstuc\expandafter{\@glo@text}}

\glsentryshortpl  Get the short plural key (as specified by the shortplural the entry was defined).
                   The argument is the label associated with the entry.
2950 \newcommand*{\glsentryshortpl}[1]{\csname glo@#1@shortpl\endcsname}

\Glsentryshortpl
2951 \newcommand*{\Glsentryshortpl}[1]{%
2952 \protected@edef\@glo@text{\csname glo@#1@shortpl\endcsname}%
2953 \expandafter\makefirstuc\expandafter{\@glo@text}}

\glsentrylong  Get the long key (as specified by the long the entry was defined). The argument
                is the label associated with the entry.
2954 \newcommand*{\glsentrylong}[1]{\csname glo@#1@long\endcsname}

\Glsentrylong
2955 \newcommand*{\Glsentrylong}[1]{%
2956 \protected@edef\@glo@text{\csname glo@#1@long\endcsname}%
2957 \expandafter\makefirstuc\expandafter{\@glo@text}}

\glsentrylongpl  Get the long plural key (as specified by the longplural the entry was defined).
                  The argument is the label associated with the entry.
2958 \newcommand*{\glsentrylongpl}[1]{\csname glo@#1@longpl\endcsname}

```

`\Glsentrylongpl`

```
2959 \newcommand*{\Glsentrylongpl}[1]{%
2960 \protected@edef\@glo@text{\csname glo@#1@longpl\endcsname}%
2961 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Short cut macros to access full form:

`\glsentryfull`

```
2962 \newcommand*{\glsentryfull}[1]{%
2963   \glsentrylong{#1}\space(\glsentryshort{#1})%
2964 }
```

`\Glsentryfull`

```
2965 \newcommand*{\Glsentryfull}[1]{%
2966   \Glsentrylong{#1}\space(\glsentryshort{#1})%
2967 }
```

`\glsentryfullpl`

```
2968 \newcommand*{\glsentryfullpl}[1]{%
2969   \glsentrylongpl{#1}\space(\glsentryshortpl{#1})%
2970 }
```

`\Glsentryfullpl`

```
2971 \newcommand*{\Glsentryfullpl}[1]{%
2972   \Glsentrylongpl{#1}\space(\glsentryshortpl{#1})%
2973 }
```

`\glsentrynumberlist` Displays the number list as is.

```
2974 \newcommand*{\glsentrynumberlist}[1]{%
2975   \glsdoifexists{#1}%
2976   {%
2977     \csname glo@#1@numberlist\endcsname
2978   }%
2979 }
```

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

```
2980 \@ifpackageloaded{hyperref}
2981 {%
2982   \newcommand*{\glsdisplaynumberlist}[1]{%
2983     \GlossariesWarning
2984     {%
2985       \string\glsdisplaynumberlist\space
2986       doesn't work with hyperref.^^JUsing
2987       \string\glsentrynumberlist\space instead%
2988     }%
2989     \glsentrynumberlist{#1}%
2990   }%
2991 }
```

```

2992 {%
2993   \newcommand*{\glsdisplaynumberlist}[1]{%
2994     \glsdoifexists{#1}%
2995     {%
2996       \bgroup
2997         \def\@glo@label{#1}%
2998         \let\@org@glnumberformat\glnumberformat
2999         \def\glnumberformat##1{##1}%
3000         \protected@edef\the@numberlist{\csname glo@\@glo@label @numberlist\endcsname}%
3001         \def\@gls@numlist@sep{}%
3002         \def\@gls@numlist@nextsep{}%
3003         \def\@gls@numlist@lastsep{}%
3004         \def\@gls@thislist{}%
3005         \def\@gls@donext@def{}%
3006         \renewcommand\do[1]{%
3007           \protected@edef\@gls@thislist{%
3008             \@gls@thislist
3009             \noexpand\@gls@numlist@sep
3010             ##1%
3011           }%
3012           \let\@gls@numlist@sep\@gls@numlist@nextsep
3013           \def\@gls@numlist@nextsep{\glnumlistsep}%
3014           \@gls@donext@def
3015           \def\@gls@donext@def{%
3016             \def\@gls@numlist@lastsep{\glnumlistlastsep}%
3017           }%
3018         }%
3019         \expandafter \glnumlistparser \expandafter{\the@numberlist}%
3020         \let\@gls@numlist@sep\@gls@numlist@lastsep
3021         \@gls@thislist
3022       \egroup
3023     }%
3024   }
3025 }

```

`\glnumlistsep`

```
3026 \newcommand*{\glnumlistsep}{, }
```

`\glnumlistlastsep`

```
3027 \newcommand*{\glnumlistlastsep}{ \& }
```

`\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.

```

3028 \newcommand*{\glshyperlink}[2][\glentrytext{\@glo@label}]{%
3029 \def\@glo@label{#2}%
3030 \@glslink{\glo@linkprefix#2}{#1}}

```

## 1.11 Adding an entry to the glossary without generating text

The following keys are provided for `\glsadd` and `\glsaddall`:

```
3031 \define@key{glossadd}{counter}{\def\@gls@counter{#1}}
```

```
3032 \define@key{glossadd}{format}{\def\@glsnumberformat{#1}}
```

This key is only used by `\glsaddall`:

```
3033 \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`

```
3034 \newrobustcmd*{\glsadd}[2][]{%
3035   \glsdoifexists{#2}%
3036   {%
3037     \def\@glsnumberformat{glsnumberformat}%
3038     \edef\@gls@counter{\csname glo@#2@counter\endcsname}%
3039     \setkeys{glossadd}{#1}%
```

Store the entry's counter in `\theglsentrycounter`

```
3040   \@gls@saveentrycounter
3041   \@do@wrglossary{#2}%
3042   }%
3043 }
```

`\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`

```
3044 \newrobustcmd*{\glsaddall}[1][]{%
3045   \edef\@glo@type{\@glo@types}%
3046   \setkeys{glossadd}{#1}%
3047   \forallglsentries[\@glo@type]{\@glo@entry}{%
3048     \glsadd[#1]{\@glo@entry}}%
3049 }
```

## 1.12 Creating associated files

The `\writeist` command creates the associated customized `.ist` makeindex style file. While defining this command, some characters have their catcodes temporarily changed to ensure they get written to the `.ist` file correctly. The

makeindex actual character (usually @) is redefined to be a ?, to allow internal commands to be written to the glossary file output file.

The special characters are stored in \@gls@actualchar, \@gls@encapchar, \@gls@levelchar and \@gls@quotechar to make them easier to use later, but don't change these values, because the characters are encoded in the command definitions that are used to escape the special characters (which means that the user no longer needs to worry about makeindex special characters).

The symbols and numbers label for group headings are hardwired into the .ist file as glssymbols and glsnumbers, the group titles can be translated (so that \glssymbolsgroupname replaces glssymbols and \glsnumbersgroupname replaces glsnumbers) using the command \glsgetgrouptitle which is defined in . This is done to prevent any problem characters in \glssymbolsgroupname and \glsnumbersgroupname from breaking hyperlinks.

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

```
3050 \edef\glsopenbrace{\expandafter\@gobble\string\{}
```

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

```
3051 \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.

```
3052 \edef\glsquote#1{\string"#1\string"}
```

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

```
3053 \ifglsxindy
3054   \newcommand*{\@glsfirstletter}{A}
3055 \fi
```

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

```
3056 \ifglsxindy
3057   \newcommand*{\GlsSetXdyFirstLetterAfterDigits}[1]{%
3058     \renewcommand*{\@glsfirstletter}{#1}}
3059 \else
3060   \newcommand*{\GlsSetXdyFirstLetterAfterDigits}[1]{%
3061     \glsnoxindywarning\GlsSetXdyFirstLetterAfterDigits}
3062 \fi
```

\@glsminrange Define the minimum number of successive location references to merge into a range.

```
3063 \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.

```
3064 \ifglsxindy
```

```

3065 \newcommand*{\GlsSetXdyMinRangeLength}[1]{%
3066 \renewcommand*{\@glsminrange}{#1}}
3067 \else
3068 \newcommand*{\GlsSetXdyMinRangeLength}[1]{%
3069 \glsnoxindywarning\GlsSetXdyMinRangeLength}
3070 \fi

```

\writeist

```

3071 \ifglxindy
    Code to use if xindy is required.
3072 \def\writeist{%
    Update attributes list
3073 \@gls@addpredefinedattributes
    Open the file.
3074 \openout\glswrite=\istfilename
    Write header comment at the start of the file
3075 \write\glswrite{;; xindy style file created by the glossaries
3076 package}%
3077 \write\glswrite{;; for document '\jobname' on
3078 \the\year-\the\month-\the\day}%
    Specify the required styles
3079 \write\glswrite{^^J; required styles^^J}
3080 \@for\@xdystyle:=\@xdyrequiredstyles\do{%
3081 \ifx\@xdystyle\@empty
3082 \else
3083 \protected@write\glswrite{{(require
3084 \string"\@xdystyle.xdy\string")}}%
3085 \fi
3086 }%
    List the allowed attributes (possible values used by the format key)
3087 \write\glswrite{^^J%
3088 ; list of allowed attributes (number formats)^^J}%
3089 \write\glswrite{(define-attributes ((\@xdyattributes)))}%
    Define any additional alphabets
3090 \write\glswrite{^^J; user defined alphabets^^J}%
3091 \write\glswrite{\@xdyuseralphabets}%
    Define location classes.
3092 \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.
3093 \@for\@gls@classI:=\@gls@xdy@locationlist\do{%

```

Case were  $\langle Hprefix \rangle$  is empty:

```

3094 \protected@write\glswrite{}\{(define-location-class
3095 \string"\@gls@classI\string"^^J\space\space\space
3096 (
3097 :sep "{\{"
3098 \csname @gls@xdy@Lclass@\@gls@classI\endcsname\space
3099 :sep "\}"
3100 )
3101 ^^J\space\space\space
3102 :min-range-length \@glsminrange^^J%
3103 )
3104 }%
```

Nested iteration over all classes:

```

3105 {%
3106 \@for\@gls@classII:=\@gls@xdy@locationlist\do{%
3107 \protected@write\glswrite{}\{(define-location-class
3108 \string"\@gls@classII-\@gls@classI\string"
3109 ^^J\space\space\space
3110 (
3111 :sep "{"
3112 \csname @gls@xdy@Lclass@\@gls@classII\endcsname\space
3113 :sep "\{"
3114 \csname @gls@xdy@Lclass@\@gls@classI\endcsname\space
3115 :sep "\}"
3116 )
3117 ^^J\space\space\space
3118 :min-range-length \@glsminrange^^J%
3119 )
3120 }%
3121 }%
3122 }%
3123 }%
```

User defined location classes (needs checking for new location format).

```

3124 \write\glswrite{^^J; user defined location classes}%
3125 \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  $\backslash\text{glsseeformat}$  which xindy won't recognise.)

```

3126 \write\glswrite{^^J; define cross-reference class^^J}%
3127 \write\glswrite{(define-crossref-class \string"see\string"
3128 :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  $\backslash\text{glsseeformat}$  which gets ignored. (When using `makeindex` this final argument contains the location information which is not required.)

```

3129 \write\glswrite{(markup-crossref-list
```

```

3130      :class \string"see\string"^^J\space\space\space
3131      :open \string"\string\glsseeformat\string"
3132      :close \string"{}\string")}%

```

List the order to sort the classes.

```

3133      \write\glswrite{^^J; define the order of the location classes}%
3134      \write\glswrite{(define-location-class-order
3135        (\@xdylocationclassorder))}%

```

Specify what to write to the start and end of the glossary file.

```

3136      \write\glswrite{^^J; define the glossary markup^^J}%

3137      \write\glswrite{(markup-index^^J\space\space\space
3138        :open \string"\string
3139        \glossarysection[\string\glossarytoctitle]{\string
3140        \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)

```

3141      \@for\@this@ctr:=\@xdycounters\do{%
3142        {%
3143          \@for\@this@attr:=\@xdyattributelist\do{%
3144            \protected\write\glswrite{{}\string\providecommand*%
3145              \expandafter\string
3146              \csname glsX\@this@ctr X\@this@attr\endcsname[2]%
3147              {%
3148                \string\setentrycounter
3149                [\expandafter\@gobble\string\#1]{\@this@ctr}%
3150                \expandafter\string
3151                \csname\@this@attr\endcsname
3152                {\expandafter\@gobble\string\#2}%
3153              }%
3154            }%
3155          }%
3156        }%
3157      }%

```

Add the end part of the open tag and the rest of the markup-index information:

```

3158      \write\glswrite{%
3159        \string\begin
3160        {theglossary}\string\glossaryheader\string~n\string" ^^J\space
3161        \space\space:close \string"\expandafter\@gobble
3162        \string%\string~n\string
3163        \end{theglossary}\string\glossarypostamble
3164        \string~n\string" ^^J\space\space\space
3165        :tree)}}%

```

Specify what to put between letter groups

```

3166      \write\glswrite{(markup-letter-group-list
3167        :sep \string"\string\glsgroupskip\string~n\string")}%

```



Specify what to put between entries

```
3168 \write\glswrite{(markup-indexentry
3169 :open \string"\string\relax \string\glsresetentrylist
3170 \string~n\string")}%
```

Specify how to format entries

```
3171 \write\glswrite{(markup-locclass-list :open
3172 \string"\glsopenbrace\string\glossaryentrynumbers
3173 \glsopenbrace\string\relax\space \string"^^J\space\space\space
3174 :sep \string", \string"
3175 :close \string"\glsclosebrace\glsclosebrace\string")}%
```

Specify how to separate location numbers

```
3176 \write\glswrite{(markup-locref-list
3177 :sep \string"\string\delimN\space\string")}%
```

Specify how to indicate location ranges

```
3178 \write\glswrite{(markup-range
3179 :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.

```
3180 \@onelevel@sanitize\gls@suffixF
3181 \@onelevel@sanitize\gls@suffixFF
3182 \ifx\gls@suffixF\@empty
3183 \else
3184 \write\glswrite{(markup-range
3185 :close "\gls@suffixF" :length 1 :ignore-end)}%
3186 \fi
3187 \ifx\gls@suffixFF\@empty
3188 \else
3189 \write\glswrite{(markup-range
3190 :close "\gls@suffixFF" :length 2 :ignore-end)}%
3191 \fi
```

Specify how to format locations.

```
3192 \write\glswrite{^^J; define format to use for locations^^J}%
3193 \write\glswrite{\@xdylocref}%
```

Specify how to separate letter groups.

```
3194 \write\glswrite{^^J; define letter group list format^^J}%
3195 \write\glswrite{(markup-letter-group-list
3196 :sep \string"\string\glsgroupskip\string~n\string")}%
```

Define letter group headings.

```
3197 \write\glswrite{^^J; letter group headings^^J}%
3198 \write\glswrite{(markup-letter-group
3199 :open-head \string"\string\glsgroupheading
3200 \glsopenbrace\string"^^J\space\space\space
3201 :close-head \string"\glsclosebrace\string")}%
```

Define additional letter groups.

```
3202 \write\glswrite{^^J; additional letter groups^^J}%
3203 \write\glswrite{\@xdylettergroups}%
```

Define additional sort rules

```
3204 \write\glswrite{^^J; additional sort rules^^J}
3205 \write\glswrite{\@xdysortrules}%
```

Close the style file

```
3206 \closeout\glswrite
```

Suppress any further calls.

```
3207 \let\writeist\relax
3208 }
3209 \else
```

Code to use if makeindex is required.

```
3210 \edef\@gls@actualchar{\string?}
3211 \edef\@gls@encapchar{\string|}
3212 \edef\@gls@levelchar{\string!}
3213 \edef\@gls@quotechar{\string"}
3214 \def\writeist{\relax
3215 \openout\glswrite=\istfilename
3216 \write\glswrite{\expandafter\@gobble\string\% makeindex style file
3217 created by the glossaries package}
3218 \write\glswrite{\expandafter\@gobble\string\% for document
3219 '\jobname' on \the\year-\the\month-\the\day}
3220 \write\glswrite{actual '\@gls@actualchar'}
3221 \write\glswrite{encap '\@gls@encapchar'}
3222 \write\glswrite{level '\@gls@levelchar'}
3223 \write\glswrite{quote '\@gls@quotechar'}
3224 \write\glswrite{keyword \string"\string\glossaryentry\string"}
3225 \write\glswrite{preamble \string"\string\glossarysection[\string
3226 \glossarytoctitle]{\string\glossarytitle}\string
3227 \glossarypreamble\string\n\string\begin{theglossary}\string
3228 \glossaryheader\string\n\string"}
3229 \write\glswrite{postamble \string"\string%\string\n\string
3230 \end{theglossary}\string\glossarypostamble\string\n
3231 \string"}
3232 \write\glswrite{group_skip \string"\string\glsgroupskip\string\n
3233 \string"}
3234 \write\glswrite{item_0 \string"\string%\string\n\string"}
3235 \write\glswrite{item_1 \string"\string%\string\n\string"}
3236 \write\glswrite{item_2 \string"\string%\string\n\string"}
3237 \write\glswrite{item_01 \string"\string%\string\n\string"}
3238 \write\glswrite{item_x1
3239 \string"\string\relax \string\glsresetentrylist\string\n
3240 \string"}
3241 \write\glswrite{item_12 \string"\string%\string\n\string"}
3242 \write\glswrite{item_x2
3243 \string"\string\relax \string\glsresetentrylist\string\n
```

```

3244     \string"}
3245     \write\glswrite{delim_0 \string"\string\{\string
3246         \glossaryentrynumbers\string\{\string\relax \string"}
3247     \write\glswrite{delim_1 \string"\string\{\string
3248         \glossaryentrynumbers\string\{\string\relax \string"}
3249     \write\glswrite{delim_2 \string"\string\{\string
3250         \glossaryentrynumbers\string\{\string\relax \string"}
3251     \write\glswrite{delim_t \string"\string\}\string\}\string"}
3252     \write\glswrite{delim_n \string"\string\delimN \string"}
3253     \write\glswrite{delim_r \string"\string\delimR \string"}
3254     \write\glswrite{headings_flag 1}
3255     \write\glswrite{heading_prefix
3256         \string"\string\glsgroupheading\string\{\string"}
3257     \write\glswrite{heading_suffix
3258         \string"\string\}\string\relax
3259         \string\glsgroupresetentrylist \string"}
3260     \write\glswrite{symhead_positive \string"glssymbols\string"}
3261     \write\glswrite{numhead_positive \string"glslnumbers\string"}
3262     \write\glswrite{page_compositor \string"glscpositor\string"}
3263     \@gls@escbsdq\gls@suffixF
3264     \@gls@escbsdq\gls@suffixFF
3265     \ifx\gls@suffixF\@empty
3266     \else
3267         \write\glswrite{suffix_2p \string"\gls@suffixF\string"}
3268     \fi
3269     \ifx\gls@suffixFF\@empty
3270     \else
3271         \write\glswrite{suffix_3p \string"\gls@suffixFF\string"}
3272     \fi
3273     \closeout\glswrite
3274     \let\writeist\relax
3275 }
3276 \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`

```

3277 \newcommand{\noist}{%
    Update attributes list
3278     \@gls@addpredefinedattributes
3279     \let\writeist\relax
3280 }

```

`\@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 T<sub>E</sub>X is trying to write to a non-existent file). The relevant glossary must be defined prior to using `\makeglossary`.

`\@makeglossary`

```
3281 \newcommand*\@makeglossary}[1]{%
3282   \ifglossaryexists{#1}%
3283   {%
```

Only create a new write if `savewrites=false` otherwise create a token to collect the information.

```
3284   \ifglssavewrites
3285     \expandafter\newtoks\csname glo@#1@filetok\endcsname
3286   \else
3287     \expandafter\newwrite\csname glo@#1@file\endcsname
3288     \expandafter\@glsopenfile\csname glo@#1@file\endcsname{#1}%
3289   \fi
3290   \@gls@renewglossary
3291   \writeist
3292 }%
3293 {%
3294   \PackageError{glossaries}%
3295   {Glossary type ‘#1’ not defined}%
3296   {New glossaries must be defined before using \string\makeglossary}%
3297 }%
3298 }
```

`\@glsopenfile` Open write file associated with the given glossary.

```
3299 \newcommand*\@glsopenfile}[2]{%
3300   \immediate\openout#1=\jobname.\csname @glotype@#2@out\endcsname
3301   \PackageInfo{glossaries}{Writing glossary file
3302     \jobname.\csname @glotype@#2@out\endcsname}%
3303 }
```

`\@nomakeglossaries` Issue warning that `\makeglossaries` hasn't been used.

```
3304 \newcommand*\@warn@nomakeglossaries{%
3305   \GlossariesWarningNoLine{\string\makeglossaries\space
3306   hasn't been used,^^Jthe glossaries will not be updated}%
3307 }
```

`\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`

```
3308 \newcommand*{\makeglossaries}{%
```

Write the name of the style file to the aux file (needed by `makeglossaries`)

```
3309 \protected@write\@auxout{}{\string\@istfilename{\istfilename}}%
```

```
3310 \protected@write\@auxout{}{\string\@glsorder{\glsorder}}%
```

Iterate through each glossary type and activate it.

```
3311 \@for\@glo@type:=\@glo@types\do{%
```

```
3312 \ifthenelse{\equal{\@glo@type}{}}{\}%
```

```
3313 \@makeglossary{\@glo@type}}%
```

```
3314 }%
```

New glossaries must be created before `\makeglossaries` so disable `\newglossary`.

```
3315 \renewcommand*\newglossary[4][]{%
```

```
3316 \PackageError{glossaries}{New glossaries
```

```
3317 must be created before \string\makeglossaries}{You need
```

```
3318 to move \string\makeglossaries\space after all your
```

```
3319 \string\newglossary\space commands}}%
```

Any subsequent instances of this command should have no effect

```
3320 \let\@makeglossary\relax
```

```
3321 \let\makeglossary\relax
```

```
3322 \let\makeglossaries\relax
```

Disable all commands that have no effect after `\makeglossaries`

```
3323 \@disable@onlypremakeg
```

Suppress warning about no `\makeglossaries`

```
3324 \let\warn@nomakeglossaries\relax
```

Declare list parser for `\glsdisplaynumberlist`

```
3325 \ifglssavenumberlist
```

```
3326 \edef\@gls@doddeflistparser{\noexpand\DeclareListParser
```

```
3327 {\noexpand\glsnumlistparser}{\delimN}}%
```

```
3328 \@gls@doddeflistparser
```

```
3329 \fi
```

```
3330 }
```

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`

```
3331 \let\makeglossary\makeglossaries
```

If `\makeglossaries` hasn't been used, issue a warning. Also issue a warning if neither `\printglossaries` nor `\printglossary` have been used.

```
3332 \AtEndDocument{%
```

```
3333 \warn@nomakeglossaries
```

```
3334 \warn@noprintglossary
```

```
3335 }
```

## 1.13 Writing information to associated files

`\glswrite` The write used for style file also used for all other output files if `savewrites=true`.

```
3336 \newwrite\glswrite
```

`\istfile` Deprecated.

```
3337 \def\istfile{\glswrite}
```

At the end of the document, the files should be created if `savewrites=true`.

```
3338 \AtEndDocument{%
```

```
3339   \glswritefiles
```

```
3340 }
```

`\glswritefiles` Only write the files if `savewrites=true`

```
3341 \ifglssavewrites
```

```
3342   \newcommand*{\glswritefiles}{%
```

Iterate through all the glossaries

```
3343   \foralllglossaries{\@glo@type}{%
```

Check for empty glossaries (patch provided by Patrick Häcker)

```
3344     \ifcsundef{glo@\@glo@type @filetok}%
```

```
3345     {%
```

```
3346       \def\gls@tmp{}%
```

```
3347     }%
```

```
3348     {%
```

```
3349       \edef\gls@tmp{\expandafter\the
```

```
3350         \csname glo@\@glo@type @filetok\endcsname}%
```

```
3351     }%
```

```
3352     \ifx\gls@tmp\@empty
```

```
3353       \ifx\@glo@type\glsdefaulttype
```

```
3354         \GlossariesWarningNoLine{Glossary ‘\@glo@type’ has no
```

```
3355           entries.^^JRemember to use package option ‘nomain’ if
```

```
3356 you
```

```
3357           don’t want to^^Juse the main glossary}%
```

```
3358       \else
```

```
3359         \GlossariesWarningNoLine{Glossary ‘\@glo@type’ has no
```

```
3360           entries}%
```

```
3361       \fi
```

```
3362     \else
```

```
3363       \@glsopenfile{\glswrite}{\@glo@type}%
```

```
3364       \immediate\write\glswrite{%
```

```
3365         \expandafter\the
```

```
3366         \csname glo@\@glo@type @filetok\endcsname}%
```

```
3367       \immediate\closeout\glswrite
```

```
3368     \fi
```

```
3369   }%
```

```
3370 }
```

```
3371 \else
```

```
3372   \let\glswritefiles\relax
```

3373 \fi

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`

```
3374 \renewcommand*{\glossary}[1][\glsdefaulttype]{%
3375   \@glossary[#1]%
3376 }
```

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`

```
3377 \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`

```
3378 \newcommand{\@gls@renewglossary}{%
3379   \gdef\@glossary[##1]{\@bsphack\begingroup\@wrglossary{##1}}%
3380   \let\@gls@renewglossary\empty
3381 }
```

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`

```
3382 \renewcommand*{\@wrglossary}[2]{%
3383   \ifglssavewrites
3384     \protected@edef\@gls@tmp{\the\csname glo@#1@filetok\endcsname#2}%
3385     \expandafter\global\expandafter\csname glo@#1@filetok\endcsname
3386       \expandafter{\@gls@tmp^^J}%
3387   \else
3388     \ifcsdef{glo@#1@file}%
3389       {%
3390         \expandafter\protected@write\csname glo@#1@file\endcsname{%
3391           \gls@disablepagerefexpansion}{#2}%
3392       }%
3393       {%
3394         \GlossariesWarning{No file defined for glossary ‘#1’}%
3395       }%
3396   \fi
```

```

3397 \endgroup\@esphack
3398 }

\@do@wrglossary
3399 \newcommand*\@do@wrglossary}[1]{%
3400 \ifglindexonlyfirst
3401 \ifglused{#1}{\@do@wrglossary{#1}}%
3402 \else
3403 \@do@wrglossary{#1}%
3404 \fi
3405 }

@protected@pagefmts List of page formats to be protected against expansion.
3406 \newcommand{\gls@protected@pagefmts}{%
3407 \gls@numberpage,\gls@alphpage,\gls@Alphpage,\gls@romanpage
3408 }

blepagerefexpansion
3409 \newcommand*\gls@disablepagerefexpansion){%
3410 \@for\@gls@this:=\gls@protected@pagefmts\do
3411 {%
3412 \expandafter\let\@gls@this\relax
3413 }%
3414 }

\gls@alphpage
3415 \newcommand*\gls@alphpage){\@alph\c@page}

\gls@Alphpage
3416 \newcommand*\gls@Alphpage){\@Alph\c@page}

\gls@numberpage
3417 \newcommand*\gls@numberpage){\number\c@page}

\gls@romanpage
3418 \newcommand*\gls@romanpage){\romannumeral\c@page}

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.
3419 \newcommand*\@do@wrglossary}[1]{%
3420 \begingroup
3421 % First a bit of hackery to prevent premature
3422 % expansion of \cs{c@page}. Store original definitions:
3423 %\changes{3.04}{2012-11-18}{modified to compensate for possible
3424 %incorrect page number}
3425 % \begin{macrocode}
3426 \let\orgnumber\number
3427 \let\orgromannumeral\romannumeral
3428 \let\orgalph\@alph
3429 \let\orgAlph\@Alph

```



Redefine:

```
3430 \def\the##1{%
3431 \ifx##1\c@page \gls@numberpage\else\orgnumber##1\fi}%
3432 \def\number##1{%
3433 \ifx##1\c@page \gls@numberpage\else\orgnumber##1\fi}%
3434 \def\romannumeral##1{%
3435 \ifx##1\c@page \gls@romanpage\else\orgnumber##1\fi}%
3436 \def\@alph##1{%
3437 \ifx##1\c@page \gls@alphpage\else\orgnumber##1\fi}%
3438 \def\@Alph##1{%
3439 \ifx##1\c@page \gls@Alphpage\else\orgnumber##1\fi}%
```

Prevent expansion:

```
3440 \gls@disablepagerefexpansion
```

Now store location in \@glslocref:

```
3441 \protected@xdef\@glslocref{\theglsentrycounter}%
3442 \endgroup
```

Escape any special characters

```
3443 \@gls@checkmkidxchars\@glslocref
```

Check if the hyper-location is the same as the location and set the hyper prefix.

```
3444 \expandafter\ifx\theHglentrycounter\theglsentrycounter
3445 \def\@glo@counterprefix{%
3446 \else
3447 \protected@edef\@glsHlocref{\theHglentrycounter}%
3448 \@gls@checkmkidxchars\@glsHlocref
3449 \edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix
3450 {\@glslocref}\@glsHlocref}%
3451 }%
3452 \@do@gls@getcounterprefix
3453 \fi
```

Determine whether to use xindy or makeindex syntax

```
3454 \ifglsxindy
```

Need to determine if the formatting information starts with a ( or ) indicating a range.

```
3455 \expandafter\@glo@check@mkidxrangechar\@glsnumberformat\@nil
3456 \def\@glo@range{%
3457 \expandafter\if\@glo@prefix(\relax
3458 \def\@glo@range{:open-range}%
3459 \else
3460 \expandafter\if\@glo@prefix)\relax
3461 \def\@glo@range{:close-range}%
3462 \fi
3463 \fi
```

Write to the glossary file using xindy syntax.

```
3464 \glossary[\csname glo@#1@type\endcsname]{%
```

```

3465 (indexentry :tkey (\csname glo@#1@index\endcsname)
3466 :locref \string{\@glo@counterprefix}{\@glslocref}\string" %
3467 :attr \string"\@gls@counter\@glo@suffix\string"
3468 \@glo@range
3469 )
3470 }%
3471 \else

```

Convert the format information into the format required for makeindex

```

3472 \@set@glo@numformat{\@glo@numfmt}{\@gls@counter}{\@glsnumberformat}%
3473 {\@glo@counterprefix}%

```

Write to the glossary file using makeindex syntax.

```

3474 \glossary[\csname glo@#1@type\endcsname]{%
3475 \string\glossaryentry{\csname glo@#1@index\endcsname
3476 \@gls@encapchar\@glo@numfmt}{\@glslocref}}%
3477 \fi
3478 }

```

`\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.)

```

3479 \newcommand*\@gls@getcounterprefix[2]{%
3480 \edef\@gls@thisloc{#1}\edef\@gls@thisHloc{#2}%
3481 \ifx\@gls@thisloc\@gls@thisHloc
3482 \def\@glo@counterprefix{}%
3483 \else
3484 \def\@gls@get@counterprefix##1.#1##2\end@getprefix{%
3485 \def\@glo@tmp{##2}%
3486 \ifx\@glo@tmp\@empty
3487 \def\@glo@counterprefix{}%
3488 \else
3489 \def\@glo@counterprefix{##1}%
3490 \fi
3491 }%
3492 \@gls@get@counterprefix#2.#1\end@getprefix
3493 \fi
3494 }

```

## 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.

```

3495 \newcommand{\@do@seeglossary}[2]{%
3496 \def\@gls@xref{#2}%

```

```

3497 \@onelevel@sanitize\@gls@xref
3498 \@gls@checkmkidxchars\@gls@xref
3499 \ifglsxindy
3500   \glossary[\csname glo@#1@type\endcsname]{%
3501     (indexentry
3502       :tkey (\csname glo@#1@index\endcsname)
3503       :xref (\string"\@gls@xref\string")
3504       :attr \string"see\string"
3505     )
3506   }%
3507 \else
3508   \glossary[\csname glo@#1@type\endcsname]{%
3509     \string\glossaryentry{\csname glo@#1@index\endcsname
3510       \@gls@encapchar glsseeformat\@gls@xref}{Z}}%
3511 \fi
3512 }

```

`\@gls@fixbraces` If no optional argument is specified, list needs to be enclosed in a set of braces.

```

3513 \def\@gls@fixbraces#1#2#3\@nil{%
3514   \ifx#2[\relax
3515     \def#1{#2#3}%
3516   \else
3517     \def#1{{#2#3}}%
3518   \fi
3519 }

```

`\glssee` `\glssee{<label>}{<cross-ref list>}`

```

3520 \newcommand*\glssee{3}[\seename]{%
3521   \@do@seeglossary{#2}{[#1]{#3}}
3522 \newcommand*\@glssee{3}[\seename]{%
3523   \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.

```

3524 \newcommand*\glsseeformat{3}[\seename]{\emph{#1} \glsseelist{#2}}

```

`\glsseelist` `\glsseelist{<list>}` formats list of entry labels.

```

3525 \newcommand*\glsseelist{1}{%

```

If there is only one item in the list, set the last separator to do nothing.

```

3526 \let\@gls@dolast\relax

```

Don’t display separator on the first iteration of the loop

```

3527 \let\@gls@donext\relax

```

Iterate through the labels

```

3528 \@for\@gls@thislabel:=#1\do{%

```

Check if on last iteration of loop

```

3529   \ifx\@xfor@nextelement\@nnil

```

```

3530      \@gls@dolast
3531      \else
3532      \@gls@donext
3533      \fi

      display the entry for this label
3534      \glsseeitem{\@gls@thislabel}%

      Update separators
3535      \let\@gls@dolast\glsseelastsep
3536      \let\@gls@donext\glsseesep
3537      }%
3538 }

```

`\glsseelastsep` Separator to use between penultimate and ultimate entries in a cross-referencing list.

```
3539 \newcommand*{\glsseelastsep}{\space\andname\space}
```

`\glsseesep` Separator to use between entires in a cross-referencing list.

```
3540 \newcommand*{\glsseesep}{, }
```

`\glsseeitem` `\glsseeitem{<label>}` formats individual entry in a cross-referencing list.

```
3541 \newcommand*{\glsseeitem}[1]{\glshyperlink[\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.)

```
3542 \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.

```

3543 \newcommand*{\gls@save@numberlist}[1]{%
3544   \ifglssavenumberlist
3545     \toks@{#1}%
3546     \edef\@do@writeaux@info{%
3547       \noexpand\csgdef{glo@\glscurrententrylabel @numberlist}{\the\toks@}%
3548     }%
3549     \@onelevel@sanitize\@do@writeaux@info
3550     \protected@write\@auxout{}\@do@writeaux@info%
3551   \fi
3552 }

```

`\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.)

```
3553 \def\warn@noprintglossary{%
3554   \GlossariesWarningNoLine{No \string\printglossary\space
3555     or \string\printglossaries\space
3556     found.^^JThis document will not have a glossary}%
3557 }
```

`\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.

```
3558 \ifcsundef{printglossary}{}%
3559 {%
```

If `\printglossary` is already defined, issue a warning and undefine it.

```
3560   \GlossariesWarning{Overriding \string\printglossary}%
3561   \undef\printglossary
3562 }
```

`\printglossary` has an optional argument. The default value is to set the glossary  
type to the main glossary.

```
3563 \newcommand*{\printglossary}[1][type=\glsdefaulttype]{%
```

If `xindy` is being used, need to find the root language for `makeglossaries` to  
pass to `xindy`.

```
3564   \ifglsxindy\findrootlanguage\fi
```

Set up defaults.

```
3565   \def\@glo@type{\glsdefaulttype}%
3566   \def\glossarytitle{\csname @glo@type\@glo@type @title\endcsname}%

3567   \def\glossarytoctitle{\glossarytitle}%
3568   \let\org@glossarytitle\glossarytitle
3569   \def\@glossarystyle{}%
3570   \def\gls@dotoc@title{\glssettoctitle{\@glo@type}}%
```

Store current value of `\glossaryentrynumbers`. (This may be changed via the  
optional argument)

```
3571   \let\org@glossaryentrynumbers\glossaryentrynumbers
```

Localise the effects of the optional argument

```
3572   \bgroup
```

Determine settings specified in the optional argument.

```
3573   \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)

```
3574   \ifx\glossarytitle\org@glossarytitle
3575   \else
```

```

3576 \expandafter\let\csname @glo@type @title\endcsname
3577 \glossarytitle
3578 \fi

```

Allow a high-level user command to indicate the current glossary

```

3579 \let\currentglossary\@glo@type

```

Enable individual number lists to be suppressed.

```

3580 \let\org@glossaryentrynumbers@glossaryentrynumbers
3581 \let\glsnonextpages\@glsnonextpages

```

Enable individual number list to be activated:

```

3582 \let\glsnextpages\@glsnextpages

```

Enable suppression of description terminators.

```

3583 \let\nopostdesc\@nopostdesc

```

Set up the entry for the TOC

```

3584 \gls@dotoc@title

```

Set the glossary style

```

3585 \@glossarystyle

```

added a way to fetch the current entry label:

```

3586 \let\gls@org@glossaryentryfield@glossaryentryfield
3587 \let\gls@org@glossarysubentryfield@glossarysubentryfield
3588 \renewcommand{\glossaryentryfield}[1]{%
3589 \gdef\glscurrententrylabel{##1}%
3590 \gls@org@glossaryentryfield{##1}%
3591 }%
3592 \renewcommand{\glossarysubentryfield}[2]{%
3593 \gdef\glscurrententrylabel{##2}%
3594 \gls@org@glossarysubentryfield{##1}{##2}%
3595 }%

```

Some macros may end up being expanded into internals in the glossary, so need to make @ a letter.

```

3596 \makeatletter

```

Input the glossary file, if it exists.

```

3597 \@input@{\jobname.\csname @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.

```

3598 \IfFileExists{\jobname.\csname @glo@type @in\endcsname}%
3599 {}%
3600 {\null}%

```

If xindy is being used, need to write the language dependent information to the .aux file for makeglossaries.

```

3601 \ifglsxindy
3602 \ifcsundef{@xdy@\@glo@type @language}%

```

```

3603     {%
3604         \edef\@do@auxoutstuff{%
3605             \noexpand\AtEndDocument{%
3606                 \noexpand\immediate\noexpand\write\@auxout{%
3607                     \string\@xdylanguage{\@glo@type}{\@xdy@main@language}}%
3608             }%
3609         }%
3610     }%
3611     {%
3612         \edef\@do@auxoutstuff{%
3613             \noexpand\AtEndDocument{%
3614                 \noexpand\immediate\noexpand\write\@auxout{%
3615                     \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type
3616                         @language\endcsname}}%
3617             }%
3618         }%
3619     }%
3620     \@do@auxoutstuff
3621     \edef\@do@auxoutstuff{%
3622         \noexpand\AtEndDocument{%
3623             \noexpand\immediate\noexpand\write\@auxout{%
3624                 \string\@gls@codepage{\@glo@type}{\@gls@codepage}}%
3625         }%
3626     }%
3627     \@do@auxoutstuff
3628     \fi
3629     \egroup

Reset \glossaryentrynumbers
3630     \global\let\glossaryentrynumbers\@org@glossaryentrynumbers

Suppress warning about no \printglossary
3631     \global\let\warn@noprintglossary\relax
3632 }

```

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`

```

3633 \newcommand*{\printglossaries}{%
3634     \forallglossaries{\@glo@type}{\printglossary[type=\@glo@type]}%
3635 }

```

The keys that can be used in the optional argument to `\printglossary` are as follows: The type key sets the glossary type.

```
3636 \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`.

```
3637 \define@key{printgloss}{title}{%
3638   \def\glossarytitle{#1}%
3639   \let\gls@dotoc@title\relax
3640 }
```

The toctitle sets the text used for the relevant entry in the table of contents.

```
3641 \define@key{printgloss}{toctitle}{%
3642   \def\glossarytoctitle{#1}%
3643   \let\gls@dotoc@title\relax
3644 }
```

The style key sets the glossary style (but only for the given glossary).

```
3645 \define@key{printgloss}{style}{%
3646   \ifcsundef{@glsstyle@#1}%
3647   {%
3648     \PackageError{glossaries}%
3649       {Glossary style ‘#1’ undefined}{}%
3650   }%
3651   {%
3652     \def\@glossarystyle{\csname @glsstyle@#1\endcsname}%
3653   }%
3654 }
```

The numberedsection key determines if this glossary should be in a numbered section.

```
3655 \define@choicekey{printgloss}{numberedsection}[\val\nr]{%
3656   false,nolabel,autolabel}[nolabel]{%
3657   \ifcase\nr\relax
3658     \renewcommand*{\@glossarysecstar}{*}%
3659     \renewcommand*{\@glossaryseclabel}{}%
3660   \or
3661     \renewcommand*{\@glossarysecstar}{}%
3662     \renewcommand*{\@glossaryseclabel}{}%
3663   \or
3664     \renewcommand*{\@glossarysecstar}{}%
3665     \renewcommand*{\@glossaryseclabel}{\label{\glsautoprefix\@glo@type}}%
3666   \fi}
```

The nonumberlist key determines if this glossary should have a number list.

```
3667 \define@boolkey{printgloss}[gls]{nonumberlist}[true]{%
3668   \ifglsnonumberlist
3669     \def\glossaryentrynumbers##1{%
3670   \else
3671     \def\glossaryentrynumbers##1{##1}%
3672   \fi}
```



`\@glsnonextpages` Suppresses the next number list only. Global assignments required as it may not occur in the same level of grouping as the next numberlist. (For example, if `\glsnonextpages` is place in the entry's description and 3 column tabular style glossary is used.) `\org@glossaryentrynumbers` needs to be set at the start of each glossary, in the event that `\glossaryentrynumber` is redefined.

```
3673 \newcommand*{\@glsnonextpages}{%
3674   \gdef\glossaryentrynumbers##1{%
3675     \glsresetentrylist
3676   }%
3677 }
```

`\@glsnextpages` Activate the next number list only. Global assignments required as it may not occur in the same level of grouping as the next numberlist. (For example, if `\glsnextpages` is place in the entry's description and 3 column tabular style glossary is used.) `\org@glossaryentrynumbers` needs to be set at the start of each glossary, in the event that `\glossaryentrynumber` is redefined.

```
3678 \newcommand*{\@glsnextpages}{%
3679   \gdef\glossaryentrynumbers##1{%
3680     ##1\glsresetentrylist}}
```

`\glsresetentrylist` Resets `\glossaryentrynumbers`

```
3681 \newcommand*{\glsresetentrylist}{%
3682   \global\let\glossaryentrynumbers\org@glossaryentrynumbers}
```

`\glsnonextpages` Outside of `\printglossary` this does nothing.

```
3683 \newcommand*{\glsnonextpages}{}%
```

`\glsnextpages` Outside of `\printglossary` this does nothing.

```
3684 \newcommand*{\glsnextpages}{}%
```

`glossaryentry` If the `entrycounter` package option has been used, define a counter to number each level 0 entry.

```
3685 \ifglentrycounter
3686   \ifx\@gls@counterwithin\@empty
3687     \newcounter{glossaryentry}
3688   \else
3689     \newcounter{glossaryentry}[\@gls@counterwithin]
3690   \fi
3691   \def\theHglossaryentry{\currentglossary.\theglossaryentry}
3692 \fi
```

`glossarysubentry` If the `subentrycounter` package option has been used, define a counter to number each level 1 entry.

```
3693 \ifglsubentrycounter
3694   \ifglentrycounter
3695     \newcounter{glossarysubentry}[glossaryentry]
3696   \else
```

```

3697   \newcounter{glossarysubentry}
3698   \fi
3699   \def\theHglossarysubentry{\currentglssubentry.\theglossarysubentry}
3700 \fi

```

**resetsubentrycounter** Resets the glossarysubentry counter.

```

3701 \ifglssubentrycounter
3702   \newcommand*{\glsresetsubentrycounter}{%
3703     \setcounter{glossarysubentry}{0}%
3704   }
3705 \else
3706   \newcommand*{\glsresetsubentrycounter}{}
3707 \fi

```

**resetsubentrycounter** Resets the glossareentry counter.

```

3708 \ifglssentrycounter
3709   \newcommand*{\glsresetentrycounter}{%
3710     \setcounter{glossaryentry}{0}%
3711   }
3712 \else
3713   \newcommand*{\glsresetentrycounter}{}
3714 \fi

```

**\glsstepentry** Advance the glossaryentry counter if in use. The argument is the label associated with the entry.

```

3715 \ifglssentrycounter
3716   \newcommand*{\glsstepentry}[1]{%
3717     \refstepcounter{glossaryentry}%
3718     \label{glssentry-#1}%
3719   }
3720 \else
3721   \newcommand*{\glsstepentry}[1]{}
3722 \fi

```

**\glsstepsubentry** Advance the glossarysubentry counter if in use. The argument is the label associated with the subentry.

```

3723 \ifglsssubentrycounter
3724   \newcommand*{\glsstepsubentry}[1]{%
3725     \def\currentglsssubentry{#1}%
3726     \refstepcounter{glossarysubentry}%
3727     \label{glsssubentry-#1}%
3728   }
3729 \else
3730   \newcommand*{\glsstepsubentry}[1]{}
3731 \fi

```

**\glsrefentry** Reference the entry or sub-entry counter if in use, otherwise just do \gls.

```

3732 \ifglssentrycounter

```

```

3733 \newcommand*{\glsrefentry}[1]{\ref{glsentry-#1}}
3734 \else
3735 \ifglssubentrycounter
3736 \newcommand*{\glsrefentry}[1]{\ref{glsentry-#1}}
3737 \else
3738 \newcommand*{\glsrefentry}[1]{\gls{#1}}
3739 \fi
3740 \fi

```

**glsentrycounterlabel** Defines how to display the glossaryentry counter.

```

3741 \ifglssentrycounter
3742 \newcommand*{\glsentrycounterlabel}{\theglossaryentry.\space}
3743 \else
3744 \newcommand*{\glsentrycounterlabel}{}
3745 \fi

```

**glsentrycounterlabel** Defines how to display the glossarysubentry counter.

```

3746 \ifglssubentrycounter
3747 \newcommand*{\glsentrycounterlabel}{\theglossarysubentry)\space}
3748 \else
3749 \newcommand*{\glsentrycounterlabel}{}
3750 \fi

```

**\glsentryitem** Step and display glossaryentry counter, if appropriate.

```

3751 \ifglssentrycounter
3752 \newcommand*{\glsentryitem}[1]{%
3753 \glsstepentry{#1}\glsentrycounterlabel
3754 }
3755 \else
3756 \newcommand*{\glsentryitem}[1]{\glsresetsubentrycounter}
3757 \fi

```

**\glssubentryitem** Step and display glossarysubentry counter, if appropriate.

```

3758 \ifglssubentrycounter
3759 \newcommand*{\glssubentryitem}[1]{%
3760 \glsstepsubentry{#1}\glssubentrycounterlabel
3761 }
3762 \else
3763 \newcommand*{\glssubentryitem}[1]{}
3764 \fi

```

**theglossary** If the theglossary environment has already been defined, a warning will be issued. This environment should be redefined by glossary styles.

```

3765 \ifcsundef{theglossary}%
3766 {%
3767 \newenvironment{theglossary}{}{}%
3768 }%
3769 {%

```

```

3770 \GlossariesWarning{overriding ‘theglossary’ environment}%
3771 \renewenvironment{theglossary}{}{}%
3772 }

```

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`

```

3773 \newcommand*{\glossaryheader}{}

```

`\glstarget` `\glstarget{<label>}{<name>}`

Provide user interface to `\@glstarget` to make it easier to modify the glossary style in the document.

```

3774 \newcommand*{\glstarget}[2]{\@glstarget{\glolinkprefix#1}{#2}}

```

`\glossaryentryfield` `\glossaryentryfield{<label>}{<name>}{<description>}{<symbol>}{<page-list>}`

This command governs how each entry row should be formatted in the glossary. Glossary styles need to redefine this command. Most of the predefined styles ignore `<symbol>`.

```

3775 \newcommand*{\glossaryentryfield}[5]{%
3776 \noindent\textbf{\glstarget{#1}{#2}} #4 #3. #5\par}

```

`\glossaryentryfield` `\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.)

```

3777 \newcommand*{\glossarysubentryfield}[6]{%
3778 \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`

```
3779 \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`

```
3780 \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.

`\glsgetgrouptitle`

```
3781 \newcommand*{\glsgetgrouptitle}[1]{%
3782   \ifcsundef{#1groupname}{#1}{\csname #1groupname\endcsname}%
3783 }
```

`\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`

```
3784 \newcommand*{\glsgetgrouplabel}[1]{%
3785   \ifthenelse{\equals{#1}{\glssymbolsgroupname}}{\glssymbols}{%
3786   \ifthenelse{\equals{#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`

```
3787 \newcommand*{\setentrycounter}[2][]{%
3788   \def\@glo@counterprefix{#1}%
3789   \ifx\@glo@counterprefix\@empty
3790     \def\@glo@counterprefix{.}%
3791   \else
3792     \def\@glo@counterprefix{.#1.}%
3793   \fi
3794   \def\glsentrycounter{#2}%
3795 }
```

The current glossary style can be set using `\glossarystyle{<style>}`.

`\glossarystyle`

```
3796 \newcommand*{\glossarystyle}[1]{%
3797   \ifcsundef{@glsstyle@#1}%
3798   {%
3799     \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
3800   }%
3801   {%
3802     \csname @glsstyle@#1\endcsname
3803   }%
3804 }
```

`\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.

```
3805 \newcommand{\newglossarystyle}[2]{%
3806   \ifcsundef{@glsstyle@#1}%
3807   {%
3808     \expandafter\def\csname @glsstyle@#1\endcsname{#2}%
3809   }%
3810   {%
3811     \PackageError{glossaries}{Glossary style ‘#1’ is already defined}{}%
3812   }%
3813 }
```

`\renewglossarystyle` Code for this macro supplied by Marco Daniel.

```
3814 \newcommand{\renewglossarystyle}[2]{%
3815   \renewcommand{\newglossarystyle}[2]{%
3816     \ifcsundef{@glsstyle@#1}%
3817     {%
3818       \expandafter\def\csname @glsstyle@#1\endcsname{#2}%
3819     }%
3820     {%
3821       \PackageError{glossaries}{Glossary style ‘#1’ is already defined}{}%
3822     }%
3823   }%
3824 }
```

```

3815 \ifcsundef{@glsstyle@#1}%
3816 {%
3817   \PackageError{glossaries}{Glossary style ‘#1’ isn’t already defined}{}%
3818 }%
3819 {%
3820   \csdef{@glsstyle@#1}{#2}%
3821 }%
3822 }

```

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`

```

3823 \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`

```

3824 \ifcsundef{hyperlink}%
3825 {%
3826   \def\glshypernumber#1{#1}%
3827 }%
3828 {%
3829   \def\glshypernumber#1{\@glshypernumber#1\nohyperpage{}}\@nil}
3830 }

```

`\@glshypernumber` This code was provided by Heiko Oberdiek to allow material to be attached to the location.

```

3831 \def\@glshypernumber#1\nohyperpage#2#3\@nil{%
3832   \ifx\#1\%
3833   \else

```

```

3834 \@@delimR#1\delimR\delimR\\%
3835 \fi
3836 \ifx\#2\\%
3837 \else
3838 #2%
3839 \fi
3840 \ifx\#3\\%
3841 \else
3842 \@gls@hypernumber#3\@nil
3843 \fi
3844 }

```

\@delimR displays a range of numbers for the counter whose name is given by \@gls@counter (which must be set prior to using \@gls@hypernumber).

\@delimR

```

3845 \def\@delimR#1\delimR #2\delimR #3\\{%
3846 \ifx\#2\\%
3847 \@delimN{#1}%
3848 \else
3849 \@gls@numberlink{#1}\delimR\@gls@numberlink{#2}%
3850 \fi}

```

\@delimN displays a list of individual numbers, instead of a range:

\@delimN

```

3851 \def\@delimN#1{\@@delimN#1\delimN \delimN\\}
3852 \def\@@delimN#1\delimN #2\delimN#3\\{%
3853 \ifx\#3\\%
3854 \@gls@numberlink{#1}%
3855 \else
3856 \@gls@numberlink{#1}\delimN\@gls@numberlink{#2}%
3857 \fi
3858 }

```

The following code is modified from hyperref's \@HyInd@pagelink where the name of the counter being used is given by \@gls@counter.

```

3859 \def\@gls@numberlink#1{%
3860 \begingroup
3861 \toks@={}%
3862 \@gls@removespaces#1 \@nil
3863 \endgroup}

3864 \def\@gls@removespaces#1 #2\@nil{%
3865 \toks@=\expandafter{\the\toks@#1}%
3866 \ifx\#2\\%
3867 \edef\x{\the\toks@}%
3868 \ifx\x\empty
3869 \else

```



```

3870     \hyperlink{\glentrycounter\glo@counterprefix\the\toks@}%
3871         {\the\toks@}%
3872     \fi
3873 \else
3874     \@gls@ReturnAfterFi{%
3875         \@gls@removespaces#2\@nil
3876     }%
3877 \fi
3878 }
3879 \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`

```
3880 \newcommand*\hyperrm[1]{\textrm{\glshypernumber{#1}}}
```

`\hypersf`

```
3881 \newcommand*\hypersf[1]{\textsf{\glshypernumber{#1}}}
```

`\hypertt`

```
3882 \newcommand*\hypertt[1]{\texttt{\glshypernumber{#1}}}
```

`\hyperbf`

```
3883 \newcommand*\hyperbf[1]{\textbf{\glshypernumber{#1}}}
```

`\hypermd`

```
3884 \newcommand*\hypermd[1]{\textmd{\glshypernumber{#1}}}
```

`\hyperit`

```
3885 \newcommand*\hyperit[1]{\textit{\glshypernumber{#1}}}
```

`\hypersl`

```
3886 \newcommand*\hypersl[1]{\textsl{\glshypernumber{#1}}}
```

`\hyperup`

```
3887 \newcommand*\hyperup[1]{\textup{\glshypernumber{#1}}}
```

`\hypersc`

```
3888 \newcommand*\hypersc[1]{\textsc{\glshypernumber{#1}}}
```

`\hyperemph`

```
3889 \newcommand*\hyperemph[1]{\emph{\glshypernumber{#1}}}
```

## 1.16 Acronyms

If the acronym package option is used, a new glossary called acronym is created

```
3890 \ifglsacronym
3891   \newglossary[alg]{acronym}{acr}{acn}{\acronymname}
      and \acronymtype is set to the name of this new glossary.
3892   \renewcommand*{\acronymtype}{acronym}
3893 \fi
```

```
\oldacronym \oldacronym[\langle label \rangle]{\langle abbrev \rangle}{\langle long \rangle}{\langle key-val list \rangle}
```

This emulates the way the old package defined acronyms. It is equivalent to `\newacronym[\langle key-val list \rangle]{\langle label \rangle}{\langle abbrev \rangle}{\langle long \rangle}` and it additionally defines the command `\langle label \rangle` which is equivalent to `\gls{\langle label \rangle}` (thus `\langle label \rangle` must only contain alphabetical characters). If `\langle label \rangle` is omitted, `\langle abbrev \rangle` 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 `\langle label \rangle` can't have an optional argument if the package is loaded. If hasn't been loaded then you can do `\langle label \rangle[\langle insert \rangle]` but you can't do `\langle label \rangle[\langle key-val list \rangle]`. 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.

```
3894 \newcommand{\oldacronym}[4][\gls@label]{%
3895   \def\gls@label{#2}%
3896   \newacronym[#4]{#1}{#2}{#3}%
3897   \ifcsundef{xspace}%
3898     {%
3899       \expandafter\edef\csname#1\endcsname{%
3900         \noexpand\@ifstar{\noexpand\Gls{#1}}{\noexpand\gls{#1}}}%
3901     }%
3902   }%
3903   {%
3904     \expandafter\edef\csname#1\endcsname{%
3905       \noexpand\@ifstar{\noexpand\Gls{#1}\noexpand\xspace}{%
3906         \noexpand\gls{#1}\noexpand\xspace}%
3907     }%
3908   }%
3909 }
```

```
\newacronym[\langle key-val list \rangle]{\langle label \rangle}{\langle abbrev \rangle}{\langle long \rangle}
```

This is a quick way of defining acronyms, all it does is call `\newglossaryentry` with the appropriate values. It sets the glossary type to `\acronymtype` which

will be acronym if the package option acronym has been used, otherwise it will be the default glossary. Since `\newacronym` merely calls `\newglossaryentry`, the acronym is treated like any other glossary entry.

If you prefer a different format, you can redefine `\newacronym` as required. The optional argument can be used to override any of the settings.

This is just a stub. It's redefined by commands like `\SetDefaultAcronymStyle`.

`\newacronym`

```
3910 \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.

```
3911 \newcommand*{\acrpluralsuffix}{\glspluralsuffix}
```

The following are defined for compatibility with version 2.07 and earlier.

`\glsshortkey`

```
3912 \newcommand*{\glsshortkey}{short}
```

`\glsshortpluralkey`

```
3913 \newcommand*{\glsshortpluralkey}{shortplural}
```

`\glslongkey`

```
3914 \newcommand*{\glslongkey}{long}
```

`\glslongpluralkey`

```
3915 \newcommand*{\glslongpluralkey}{longplural}
```

`\acrfull` Full form of the acronym.

```
3916 \newrobustcmd*{\acrfull}{%
```

```
3917 \@ifstar\s@acrfull\ns@acrfull
```

```
3918 }
```

```
3919 \newcommand*\s@acrfull[2] [] {%
```

```
3920 \new@ifnextchar[{\@acrfull{hyper=false,#1}{#2}}}%
```

```
3921 \@acrfull{hyper=false,#1}{#2} []}%
```

```
3922 }
```

```
3923 \newcommand*\ns@acrfull[2] [] {%
```

```
3924 \new@ifnextchar[{\@acrfull{#1}{#2}}}%
```

```
3925 \@acrfull{#1}{#2} []}%
```

```
3926 }
```

Low-level macro:

```
3927 \def\@acrfull#1#2[#3]{%
3928   \acrlinkfullformat{\@acrlong}{\@acrshort}{#1}{#2}{#3}%
3929 }
```

```
\acrlinkfullformat  Format for full links like \acrfull.  Syntax: \acrlinkfullformat{<long
                    cs>}{<short cs>}{<options>}{<label>}{<insert>}
3930 \newcommand{\acrlinkfullformat}[5]{%
3931   \acrfullformat{#1{#3}{#4}[#5]}{#2{#3}{#4}[]}%
3932 }
```

\acrfullformat Default full form is <long> (<short>).

```
3933 \newcommand{\acrfullformat}[2]{#1\space(#2)}
```

Default format for full acronym

\Acrfull

```
3934 \newrobustcmd*{\Acrfull}{%
3935   \@ifstar\s@Acrfull\@ns@Acrfull
3936 }

3937 \newcommand*\s@Acrfull[2][]{%
3938   \new@ifnextchar[{\@Acrfull{hyper=false,#1}{#2}}%
3939     {\@Acrfull{hyper=false,#1}{#2}[]}%
3940 }
3941 \newcommand*\ns@Acrfull[2][]{%
3942   \new@ifnextchar[{\@Acrfull{#1}{#2}}%
3943     {\@Acrfull{#1}{#2}[]}%
3944 }
```

Low-level macro:

```
3945 \def\@Acrfull#1#2[#3]{%
3946   \acrlinkfullformat{\@Acrlong}{\@acrshort}{#1}{#2}{#3}%
3947 }
```

\ACRfull

```
3948 \newrobustcmd*{\ACRfull}{%
3949   \@ifstar\s@ACRfull\@ns@ACRfull
3950 }

3951 \newcommand*\s@ACRfull[2][]{%
3952   \new@ifnextchar[{\@ACRfull{hyper=false,#1}{#2}}%
3953     {\@ACRfull{hyper=false,#1}{#2}[]}%
3954 }
3955 \newcommand*\ns@ACRfull[2][]{%
3956   \new@ifnextchar[{\@ACRfull{#1}{#2}}%
3957     {\@ACRfull{#1}{#2}[]}%
3958 }
```

Low-level macro:

```
3959 \def\@ACRfull#1#2[#3]{%
3960   \acrlinkfullformat{\@ACRlong}{\@ACRshort}{#1}{#2}{#3}%
3961 }
```

Plural:

\acrfullpl

```
3962 \newrobustcmd*\acrfullpl{%
3963   \@ifstar\s@acrfullpl\ns@acrfullpl
3964 }

3965 \newcommand*\s@acrfullpl[2][{}]{%
3966   \new@ifnextchar[{\@acrfullpl{hyper=false,#1}{#2}}%
3967     {\@acrfullpl{hyper=false,#1}{#2}[]}%
3968 }
3969 \newcommand*\ns@acrfullpl[2][{}]{%
3970   \new@ifnextchar[{\@acrfullpl{#1}{#2}}%
3971     {\@acrfullpl{#1}{#2}[]}%
3972 }
```

Low-level macro:

```
3973 \def\@acrfullpl#1#2[#3]{%
3974   \acrlinkfullformat{\@acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%
3975 }
```

\Acrfullpl

```
3976 \newrobustcmd*\Acrfullpl{%
3977   \@ifstar\s@Acrfullpl\ns@Acrfullpl
3978 }

3979 \newcommand*\s@Acrfullpl[2][{}]{%
3980   \new@ifnextchar[{\@Acrfullpl{hyper=false,#1}{#2}}%
3981     {\@Acrfullpl{hyper=false,#1}{#2}[]}%
3982 }
3983 \newcommand*\ns@Acrfullpl[2][{}]{%
3984   \new@ifnextchar[{\@Acrfullpl{#1}{#2}}%
3985     {\@Acrfullpl{#1}{#2}[]}%
3986 }
```

Low-level macro:

```
3987 \def\@Acrfullpl#1#2[#3]{%
3988   \acrlinkfullformat{\@Acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%
3989 }
```

\ACRfullpl

```
3990 \newrobustcmd*\ACRfullpl{%
3991   \@ifstar\s@ACRfullpl\ns@ACRfullpl
3992 }
```

```

3993 \newcommand*\s@ACRfullpl[2] [] {%
3994   \new@ifnextchar[{\@ACRfullpl{hyper=false,#1}{#2}}{%
3995     {\@ACRfullpl{hyper=false,#1}{#2} []}%
3996 }
3997 \newcommand*\ns@ACRfullpl[2] [] {%
3998   \new@ifnextchar[{\@ACRfullpl{#1}{#2}}{%
3999     {\@ACRfullpl{#1}{#2} []}%
4000 }

```

Low-level macro:

```

4001 \def\@ACRfullpl#1#2[#3] {%
4002   \acrlinkfullformat{\@ACRlongpl}{\@ACRshortpl}{#1}{#2}{#3}%
4003 }

```

## 1.17 Predefined acronym styles

`\acronymfont` This is only used with the additional acronym styles:

```

4004 \newcommand{\acronymfont}[1]{#1}

```

`\firstacronymfont` This is only used with the additional acronym styles:

```

4005 \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.

```

4006 \newcommand*\acrnameformat[2]{\acronymfont{#1}}

```

Define some tokens used by `\newacronym`:

`\glskeylisttok`

```

4007 \newtoks\glskeylisttok

```

`\glslabeltok`

```

4008 \newtoks\glslabeltok

```

`\glsshorttok`

```

4009 \newtoks\glsshorttok

```

`\glslongtok`

```

4010 \newtoks\glslongtok

```

`\newacronymhook` Provide a hook for `\newacronym`:

```

4011 \newcommand*\newacronymhook{}

```

`AcronymDisplayStyle` Sets the default acronym display style for given glossary.

```

4012 \newcommand*\SetDefaultAcronymDisplayStyle[1] {%
4013   \defglsdisplay[#1]{##1##4}%
4014   \defglsdisplayfirst[#1]{##1##4}%
4015 }

```

`defaultNewAcronymDef` Sets up the acronym definition for the default style. The information is provided by the tokens `\glslabeltok`, `\glsshorttok`, `\glslongtok` and `\glskeylisttok`.

```

4016 \newcommand*\DefaultNewAcronymDef{%
4017   \edef\@do@newglossaryentry{%
4018     \noexpand\newglossaryentry{\the\glslabeltok}%
4019     {%
4020       type=\acronymtype,%
4021       name={\the\glsshorttok},%
4022       sort={\the\glsshorttok},%
4023       text={\the\glsshorttok},%
4024       first={\acrfullformat{\the\glslongtok}{\the\glsshorttok}},%
4025       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4026       firstplural={\acrfullformat{\noexpand\@glo@longpl}%
4027                     {\noexpand\@glo@shortpl}},%
4028       short={\the\glsshorttok},%
4029       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4030       long={\the\glslongtok},%
4031       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4032       description={\the\glslongtok},%
4033       descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%

```

Remaining options specified by the user:

```

4034     \the\glskeylisttok
4035   }%
4036 }%
4037 \@do@newglossaryentry
4038 }

```

`DefaultAcronymStyle` Set up the default acronym style:

```

4039 \newcommand*\SetDefaultAcronymStyle{%
  Set the display style:
4040   \@for\@gls@type:=\@glsacronymlists\do{%
4041     \SetDefaultAcronymDisplayStyle{\@gls@type}%
4042   }%

```

Set up the definition of `\newacronym`:

```

4043 \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).

```

4044   \ifx\@glsacronymlists\@empty
4045     \def\@glo@type{\acronymtype}%
4046     \setkeys{glossentry}{##1}%
4047     \DeclareAcronymList{\@glo@type}%
4048     \SetDefaultAcronymDisplayStyle{\@glo@type}%
4049   \fi
4050   \glskeylisttok{##1}%

```

```

4051 \glslabeltok{##2}%
4052 \glsshorttok{##3}%
4053 \glslongtok{##4}%
4054 \newacronymhook
4055 \DefaultNewAcronymDef
4056 }%
4057 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
4058 }

```

`\acrfootnote` Used by the footnote acronym styles.

```

4059 \newcommand*{\acrfootnote}[3]{\acrlinkfootnote{#1}{#2}{#3}}

```

`\acrlinkfootnote`

```

4060 \newcommand*{\acrlinkfootnote}[3]{%
4061 \footnote{\glslink[1]{#2}{#3}}%
4062 }

```

`\acrnofootnote`

```

4063 \newcommand*{\acrnofootnote}[3]{%
4064 \footnote{#3}%
4065 }

```

`AcronymDisplayStyle` Sets the acronym display style for given glossary for the description and footnote combination.

```

4066 \newcommand*{\SetDescriptionFootnoteAcronymDisplayStyle}[1]{%
4067 \defglsdisplayfirst[#1]{%
4068 \firstacronymfont{##1}##4%
4069 \expandafter\protect\expandafter\acrfootnote\expandafter
4070 {\@gls@link@opts}{\@gls@link@label}{##3}%
4071 }%
4072 \defglsdisplay[#1]{\acronymfont{##1}##4}%
4073 }

```

`otnoteNewAcronymDef`

```

4074 \newcommand*{\DescriptionFootnoteNewAcronymDef}{%
4075 \edef\@do@newglossaryentry{%
4076 \noexpand\newglossaryentry{\the\glslabeltok}%
4077 {%
4078 type=\acronymtype,%
4079 name={\noexpand\acronymfont{\the\glsshorttok}},%
4080 sort={\the\glsshorttok},%
4081 text={\the\glsshorttok},%
4082 plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4083 short={\the\glsshorttok},%
4084 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4085 long={\the\glslongtok},%
4086 longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4087 symbol={\the\glslongtok},%

```



```

4088         symbolplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4089         \the\glskeylisttok
4090     }%
4091 }%
4092 \do@newglossaryentry
4093 }

```

**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.

```

4094 \newcommand*\SetDescriptionFootnoteAcronymStyle{%
4095     \renewcommand{\newacronym}[4][\]{%
4096         \ifx\@glsacronymlists\@empty
4097             \def\@glo@type{\acronymtype}%
4098             \setkeys{glossentry}{##1}%
4099             \DeclareAcronymList{\@glo@type}%
4100             \SetDescriptionFootnoteAcronymDisplayStyle{\@glo@type}%
4101         \fi
4102         \glskeylisttok{##1}%
4103         \glslabeltok{##2}%
4104         \glsshorttok{##3}%
4105         \glslongtok{##4}%
4106         \newacronymhook
4107         \DescriptionFootnoteNewAcronymDef
4108     }%

```

If footnote package option is specified, set the first use to append the long form (stored in symbol) as a footnote.

```

4109     \@for\@gls@type:=\@glsacronymlists\do{%
4110         \SetDescriptionFootnoteAcronymDisplayStyle{\@gls@type}%
4111     }%

```

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.

```

4112     \ifglsacrsmallcaps
4113         \renewcommand*\acronymfont[1]{\textsc{##1}}%
4114         \renewcommand*\acrpluralsuffix{%
4115             \textup{\glspluralsuffix}}%
4116     \else
4117         \ifglsacrsmaller
4118             \renewcommand*\acronymfont[1]{\textsmaller{##1}}%
4119         \fi
4120     \fi

```

Check for package option clash

```

4121     \ifglsacrdua
4122         \PackageError{glossaries}{Option clash: ‘footnote’ and ‘dua’
4123             can’t both be set}{\}%

```

```

4124 \fi
4125 }%

```

**AcronymDisplayStyle** Sets the acronym display style for given glossary with description and dua combination.

```

4126 \newcommand*{\SetDescriptionDUAAcronymDisplayStyle}[1]{%
4127   \defglstdisplay[#1]{##1##4}%
4128   \defglstdisplayfirst[#1]{##1##4}%
4129 }

```

**ionDUANewAcronymDef**

```

4130 \newcommand*{\DescriptionDUANewAcronymDef}{%
4131   \edef\@do@newglossaryentry{%
4132     \noexpand\newglossaryentry{\the\glslabeltok}%
4133     {%
4134       type=\acronymtype,%
4135       name={\the\glslongtok},%
4136       sort={\the\glslongtok},%
4137       text={\the\glslongtok},%
4138       plural={\the\glslongtok\noexpand\acrpluralsuffix},%
4139       short={\the\glsshorttok},%
4140       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4141       long={\the\glslongtok},%
4142       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4143       symbol={\the\glsshorttok},%
4144       symbolplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4145       \the\glskeylisttok
4146     }%
4147   }%
4148   \@do@newglossaryentry
4149 }

```

**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.

```

4150 \newcommand*{\SetDescriptionDUAAcronymStyle}{%
4151   \ifglssacrsmallcaps
4152     \PackageError{glossaries}{Option clash: 'smallcaps' and 'dua'
4153       can't both be set}{}%
4154   \else
4155     \ifglssacrsmaller
4156       \PackageError{glossaries}{Option clash: 'smaller' and 'dua'
4157         can't both be set}{}%
4158     \fi
4159   \fi
4160   \renewcommand{\newacronym}[4][[]]{%
4161     \ifx\@glsacronymlists\@empty
4162       \def\@glo@type{\acronymtype}%
4163       \setkeys{glossentry}{##1}%

```

```

4164     \DeclareAcronymList{\@glo@type}%
4165     \SetDescriptionDUAAcronymDisplayStyle{\@glo@type}%
4166     \fi
4167     \glskeylisttok{##1}%
4168     \glslabeltok{##2}%
4169     \glsshorttok{##3}%
4170     \glslongtok{##4}%
4171     \newacronymhook
4172     \DescriptionDUANewAcronymDef
4173 }%

Set display.
4174 \@for\@gls@type:=\@glsacronymlists\do{%
4175     \SetDescriptionDUAAcronymDisplayStyle{\@gls@type}%
4176 }%
4177}%

```

**AcronymDisplayStyle** Sets the acronym display style for given glossary using the description setting (but not footnote or dua).

```

4178 \newcommand*\SetDescriptionAcronymDisplayStyle[1]{%
4179     \defglsdisplayfirst[#1]{%
4180         ##1##4 (\firstacronymfont{##3})}%
4181     \defglsdisplay[#1]{\acronymfont{##1}##4}%
4182 }

```

**DescriptionNewAcronymDef**

```

4183 \newcommand*\DescriptionNewAcronymDef{%
4184     \edef\@do@newglossaryentry{%
4185         \noexpand\newglossaryentry{\the\glslabeltok}%
4186         {%
4187             type=\acronymtype,%
4188             name={\noexpand
4189                 \acrnameformat{\the\glsshorttok}{\the\glslongtok}},%
4190             sort={\the\glsshorttok},%
4191             first={\the\glslongtok},%
4192             firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4193             text={\the\glsshorttok},%
4194             plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4195             short={\the\glsshorttok},%
4196             shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4197             long={\the\glslongtok},%
4198             longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4199             symbol={\noexpand\@glo@text},%
4200             symbolplural={\noexpand\@glo@plural},%
4201             \the\glskeylisttok}%
4202     }%
4203     \@do@newglossaryentry
4204 }

```

`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.

```

4205 \newcommand*{\SetDescriptionAcronymStyle}{%
4206   \renewcommand{\newacronym}[4][\]{%
4207     \ifx\@glsacronymlists\@empty
4208       \def\@glo@type{\acronymtype}%
4209       \setkeys{glossentry}{##1}%
4210       \DeclareAcronymList{\@glo@type}%
4211       \SetDescriptionAcronymDisplayStyle{\@glo@type}%
4212     \fi
4213     \glskeylisttok{##1}%
4214     \glslabeltok{##2}%
4215     \glsshorttok{##3}%
4216     \gslongtok{##4}%
4217     \newacronymhook
4218     \DescriptionNewAcronymDef
4219   }%

```

Set display.

```

4220 \@for\@gls@type:=\@glsacronymlists\do{%
4221   \SetDescriptionAcronymDisplayStyle{\@gls@type}%
4222 }%

```

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.

```

4223 \ifglsacrsmallcaps
4224   \renewcommand{\acronymfont}[1]{\textsc{##1}}
4225   \renewcommand*{\acrpluralsuffix}{%
4226     \textup{\glspluralsuffix}}%
4227 \else
4228   \ifglsacrsmaller
4229     \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%
4230   \fi
4231 \fi
4232 }%

```

`AcronymDisplayStyle` Sets the acronym display style for given glossary with footnote setting (but not description or dua).

```

4233 \newcommand*{\SetFootnoteAcronymDisplayStyle}[1]{%
4234   \defglsdisplayfirst[#1]{%
4235     \firstacronymfont{##1}##4%
4236     \expandafter\protect\expandafter\acrfootnote\expandafter
4237       {\@gls@link@opts}{\@gls@link@label}{##2}%
4238   }%
4239   \defglsdisplay[#1]{\acronymfont{##1}##4}%
4240 }

```

otnoteNewAcronymDef

```

4241 \newcommand*{\FootnoteNewAcronymDef}{%
4242   \edef\@do@newglossaryentry{%
4243     \noexpand\newglossaryentry{\the\glslabeltok}%
4244     {%
4245       type=\acronymtype,%
4246       name={\noexpand\acronymfont{\the\glsshorttok}},%
4247       sort={\the\glsshorttok},%
4248       text={\the\glsshorttok},%
4249       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4250       short={\the\glsshorttok},%
4251       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4252       long={\the\glslongtok},%
4253       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4254       description={\the\glslongtok},%
4255       descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4256       \the\glskeylisttok
4257     }%
4258   }%
4259   \@do@newglossaryentry
4260 }

```

otnoteAcronymStyle 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.

```

4261 \newcommand*{\SetFootnoteAcronymStyle}{%
4262   \renewcommand{\newacronym}[4][\]{%
4263     \ifx\@glsacronymlists\@empty
4264       \def\@glo@type{\acronymtype}%
4265       \setkeys{glossentry}{##1}%
4266       \DeclareAcronymList{\@glo@type}%
4267       \SetFootnoteAcronymDisplayStyle{\@glo@type}%
4268     \fi
4269     \glskeylisttok{##1}%
4270     \glslabeltok{##2}%
4271     \glsshorttok{##3}%
4272     \glslongtok{##4}%
4273     \newacronymhook
4274     \FootnoteNewAcronymDef
4275   }%

```

Set display

```

4276   \@for\@gls@type:=\@glsacronymlists\do{%
4277     \SetFootnoteAcronymDisplayStyle{\@gls@type}%
4278   }%

```

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.

```

4279 \ifglsmallcaps
4280 \renewcommand*{\acronymfont}[1]{\textsc{##1}}%
4281 \renewcommand*{\acrpluralsuffix}{%
4282 \textup{\glpluralsuffix}}%
4283 \else
4284 \ifglsmaller
4285 \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%
4286 \fi
4287 \fi

Check for option clash

4288 \ifglacrdua
4289 \PackageError{glossaries}{Option clash: ‘footnote’ and ‘dua’
4290 can’t both be set}{}%
4291 \fi
4292 }%

```

**AcronymDisplayStyle** Sets the acronym display style for given glossary where neither footnote nor description is required, but smallcaps or smaller specified.

```

4293 \newcommand*{\SetSmallAcronymDisplayStyle}[1]{%
4294 \defglfirst[#1]{##1##4 (\firstacronymfont{##3})}
4295 \defglsecond[#1]{\acronymfont{##1}##4}%
4296 }

```

**\SmallNewAcronymDef**

```

4297 \newcommand*{\SmallNewAcronymDef}{%
4298 \edef\@do@newglossaryentry{%
4299 \noexpand\newglossaryentry{\the\glslabelltok}%
4300 {%
4301 type=\acronymtype,%
4302 name={\noexpand\acronymfont{\the\glsshorttok}},%
4303 sort={\the\glsshorttok},%
4304 text={\noexpand\@glo@symbol},%

```

Default to the short plural.

```

4305 plural={\noexpand\@glo@shortpl},%
4306 first={\the\glslongtok},%

```

Default to the long plural.

```

4307 firstplural={\noexpand\@glo@longpl},%
4308 short={\the\glsshorttok},%
4309 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4310 long={\the\glslongtok},%
4311 longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4312 description={\noexpand\@glo@first},%
4313 descriptionplural={\noexpand\@glo@firstplural},%
4314 symbol={\the\glsshorttok},%

```

Default to the short plural.

```

4315 symbolplural={\noexpand\@glo@shortpl},%
4316 \the\glkeylisttok

```

```

4317 }%
4318 }%
4319 \@do@newglossaryentry
4320 }

```

`\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.

```

4321 \newcommand*\setSmallAcronymStyle{%
4322   \renewcommand{\newacronym}[4][\]{%
4323     \ifx\@glsacronymlists\@empty
4324       \def\@glo@type{\acronymtype}%
4325       \setkeys{glossentry}{##1}%
4326       \DeclareAcronymList{\@glo@type}%
4327       \setSmallAcronymDisplayStyle{\@glo@type}%
4328     \fi
4329     \glskeylisttok{##1}%
4330     \glslabeltok{##2}%
4331     \glsshorttok{##3}%
4332     \glslongtok{##4}%
4333     \newacronymhook
4334     \SmallNewAcronymDef
4335   }%

```

Change the display since first only contains long form.

```

4336   \@for\@gls@type:=\@glsacronymlists\do{%
4337     \setSmallAcronymDisplayStyle{\@gls@type}%
4338   }%

```

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.

```

4339   \ifglsacrsmallcaps
4340     \renewcommand*\acronymfont[1]{\textsc{##1}}
4341     \renewcommand*\acrpluralsuffix{%
4342       \textup{\glspluralsuffix}}%
4343   \else
4344     \renewcommand*\acronymfont[1]{\textsmaller{##1}}
4345   \fi

```

check for option clash

```

4346   \ifglsacrdua
4347     \ifglsacrsmallcaps
4348       \PackageError{glossaries}{Option clash: 'smallcaps' and 'dua'
4349         can't both be set}{}%
4350     \else
4351       \PackageError{glossaries}{Option clash: 'smaller' and 'dua'
4352         can't both be set}{}%
4353     \fi
4354   \fi
4355 }%

```

`\SetDUADisplayStyle` Sets the acronym display style for given glossary with dua setting.

```

4356 \newcommand*\SetDUADisplayStyle}[1]{%
4357   \defglsdisplay[#1]{##1##4}%
4358   \defglsdisplayfirst[#1]{##1##4}%
4359 }

```

`\DUANewAcronymDef`

```

4360 \newcommand*\DUANewAcronymDef{%
4361   \edef\@do@newglossaryentry{%
4362     \noexpand\newglossaryentry{\the\glslabeltok}%
4363     {%
4364       type=\acronymtype,%
4365       name={\the\glsshorttok},%
4366       text={\the\glslongtok},%
4367       plural={\the\glslongtok\noexpand\acrpluralsuffix},%
4368       short={\the\glsshorttok},%
4369       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4370       long={\the\glslongtok},%
4371       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4372       description={\the\glslongtok},%
4373       symbol={\the\glsshorttok},%
4374       symbolplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4375       \the\glskeylisttok
4376     }%
4377   }%
4378   \@do@newglossaryentry
4379 }

```

`\SetDUAStyle` Always expand acronyms.

```

4380 \newcommand*\SetDUAStyle{%
4381   \renewcommand{\newacronym}[4][\]{%
4382     \ifx\@glsacronymlists\@empty
4383       \def\@glo@type{\acronymtype}%
4384       \setkeys{glossentry}{##1}%
4385       \DeclareAcronymList{\@glo@type}%
4386       \SetDUADisplayStyle{\@glo@type}%
4387     \fi
4388     \glskeylisttok{##1}%
4389     \glslabeltok{##2}%
4390     \glsshorttok{##3}%
4391     \glslongtok{##4}%
4392     \newacronymhook
4393     \DUANewAcronymDef
4394   }%

   Set the display
4395   \@for\@gls@type:=\@glsacronymlists\do{%
4396     \SetDUADisplayStyle{\@gls@type}%
4397   }%
4398 }

```



\SetAcronymStyle

```
4399 \newcommand*{\SetAcronymStyle}{%
4400   \SetDefaultAcronymStyle
4401   \ifglssacrdescription
4402     \ifglssacrfootnote
4403       \SetDescriptionFootnoteAcronymStyle
4404     \else
4405       \ifglssacrdua
4406         \SetDescriptionDUAAcronymStyle
4407       \else
4408         \SetDescriptionAcronymStyle
4409       \fi
4410     \fi
4411   \else
4412     \ifglssacrfootnote
4413       \SetFootnoteAcronymStyle
4414     \else
4415       \ifthenelse{\boolean{glssacrsmallsac}s}{OR
4416         \boolean{glssacrsmallsac}}{%
4417         {%
4418           \SetSmallAcronymStyle
4419         }%
4420         {%
4421           \ifglssacrdua
4422             \SetDUASStyle
4423           \fi
4424         }%
4425       \fi
4426     \fi
4427 }
```

Set the acronym style according to the package options

4428 \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.

tCustomDisplayStyle Sets the acronym display style.

```
4429 \newcommand*{\SetCustomDisplayStyle}[1]{%
4430   \defglssdisplay[#1]{##1##4}%
4431   \defglssdisplayfirst[#1]{##1##4}%
4432 }
```

CustomAcronymFields

```
4433 \newcommand*{\CustomAcronymFields}{%
4434   name={\the\glssshorttok},%
```

```

4435 description={\the\glslongtok},%
4436 first={\noexpand\acrfullformat{\the\glslongtok}{\the\glsshorttok}},%
4437 firstplural={\noexpand\acrfullformat
4438   {\the\glslongtok\noexpand\acrpluralsuffix}{\the\glsshorttok}}%
4439 text={\the\glsshorttok},%
4440 plural={\the\glsshorttok\noexpand\acrpluralsuffix}%
4441 }

```

#### CustomNewAcronymDef

```

4442 \newcommand*{\CustomNewAcronymDef}{%
4443   \protected@edef\@do@newglossaryentry{%
4444     \noexpand\newglossaryentry{\the\glslabeltok}%
4445     {%
4446       type=\acronymtype,%
4447       short={\the\glsshorttok},%
4448       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4449       long={\the\glslongtok},%
4450       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4451       user1={\the\glsshorttok},%
4452       user2={\the\glsshorttok\noexpand\acrpluralsuffix},%
4453       user3={\the\glslongtok},%
4454       user4={\the\glslongtok\noexpand\acrpluralsuffix},%
4455       \CustomAcronymFields,%
4456       \the\glskeylisttok
4457     }%
4458   }%
4459   \@do@newglossaryentry
4460 }

```

#### \SetCustomStyle

```

4461 \newcommand*{\SetCustomStyle}{%
4462   \renewcommand{\newacronym}[4][\]{%
4463     \ifx\@glsacronymlists\@empty
4464       \def\@glo@type{\acronymtype}%
4465       \setkeys{glossentry}{##1}%
4466       \DeclareAcronymList{\@glo@type}%
4467       \SetCustomDisplayStyle{\@glo@type}%
4468     \fi
4469     \glskeylisttok{##1}%
4470     \glslabeltok{##2}%
4471     \glsshorttok{##3}%
4472     \glslongtok{##4}%
4473     \newacronymhook
4474     \CustomNewAcronymDef
4475   }%

```

#### Set the display

```

4476   \@for\@gls@type:=\@glsacronymlists\do{%
4477     \SetCustomDisplayStyle{\@gls@type}%
4478   }%

```

4479 }

fineAcronymSynonyms

4480 \newcommand\*{\DefineAcronymSynonyms}{%

Short form

\acs

4481 \let\acs\acrshort

First letter uppercase short form

\Acs

4482 \let\Acs\Acrshort

Plural short form

\acsp

4483 \let\acsp\acrshortpl

First letter uppercase plural short form

\Acsp

4484 \let\Acsp\Acrshortpl

Long form

\acl

4485 \let\acl\aclong

Plural long form

\aclp

4486 \let\aclp\aclongpl

First letter upper case long form

\Acl

4487 \let\Acl\Aclong

First letter upper case plural long form

\Aclp

4488 \let\Aclp\Aclongpl

Full form

\acf

4489 \let\acf\acrfull

Plural full form

```

\acfp
4490 \let\acfp\acrfullpl

First letter upper case full form

\Acf
4491 \let\Acf\Acrfull

First letter upper case plural full form

\Acfp
4492 \let\Acfp\Acrfullpl

Standard form

\ac
4493 \let\ac\gls

First upper case standard form

\Ac
4494 \let\Ac\Gls

Standard plural form

\acp
4495 \let\acp\glspl

Standard first letter upper case plural form

\Acp
4496 \let\Acp\Glspl
4497 }

Define synonyms if required
4498 \ifglsacrshortcuts
4499 \DefineAcronymSynonyms
4500 \fi

```

## 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.

```
4501 \RequirePackage{glossary-hypernav}
```

The styles that use list-like environments. These are not loaded if the `nolist` option is used:

```
4502 \@gls@loadlist
```

The styles that use the longtable environment. These are not loaded if the no-long package option is used.

```
4503 \@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.

```
4504 \@gls@loadsuper
```

The tree-like styles. These are not loaded if the notree package option is used.

```
4505 \@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.

```
4506 \ifx\@glossary@default@style\relax
4507 \else
4508   \glossarystyle{\@glossary@default@style}
4509 \fi
```

## 1.19 Debugging Commands

\showgloparent    \showgloparent{<label>}

```
4510 \newcommand*\showgloparent}[1]{%
4511   \expandafter\show\csname glo@#1@parent\endcsname
4512 }
```

\showglolevel    \showglolevel{<label>}

```
4513 \newcommand*\showglolevel}[1]{%
4514   \expandafter\show\csname glo@#1@level\endcsname
4515 }
```

\showglotext    \showglotext{<label>}

```
4516 \newcommand*\showglotext}[1]{%
4517   \expandafter\show\csname glo@#1@text\endcsname
4518 }
```

\showgloplural    \showgloplural{<label>}

```
4519 \newcommand*\showgloplural}[1]{%
4520   \expandafter\show\csname glo@#1@plural\endcsname
4521 }
```

\showglofirst    \showglofirst{<label>}

```
4522 \newcommand*{\showglofirst}[1]{%
4523   \expandafter\show\csname glo@#1@first\endcsname
4524 }
```

\showglofirstpl    \showglofirstpl{<label>}

```
4525 \newcommand*{\showglofirstpl}[1]{%
4526   \expandafter\show\csname glo@#1@firstpl\endcsname
4527 }
```

\showglotype    \showglotype{<label>}

```
4528 \newcommand*{\showglotype}[1]{%
4529   \expandafter\show\csname glo@#1@type\endcsname
4530 }
```

\showglocounter    \showglocounter{<label>}

```
4531 \newcommand*{\showglocounter}[1]{%
4532   \expandafter\show\csname glo@#1@counter\endcsname
4533 }
```

\showglouser    \showglouser{<label>}

```
4534 \newcommand*{\showglouser}[1]{%
4535   \expandafter\show\csname glo@#1@useri\endcsname
4536 }
```

\showglouserii    \showglouserii{<label>}

```
4537 \newcommand*{\showglouserii}[1]{%
4538   \expandafter\show\csname glo@#1@userii\endcsname
4539 }
```

\showglouseriii    \showglouseriii{<label>}

```
4540 \newcommand*{\showglouseriii}[1]{%
4541   \expandafter\show\csname glo@#1@useriii\endcsname
4542 }
```

\showglouseriv    \showglouseriv{<label>}

```
4543 \newcommand*{\showglouseriv}[1]{%
4544   \expandafter\show\csname glo@#1@useriv\endcsname
4545 }
```

\showglouserv    \showglouserv{<label>}

```
4546 \newcommand*{\showglouserv}[1]{%
4547   \expandafter\show\csname glo@#1@userv\endcsname
4548 }
```

\showglouservi    \showglouservi{<label>}

```
4549 \newcommand*{\showglouservi}[1]{%
4550   \expandafter\show\csname glo@#1@uservi\endcsname
4551 }
```

\showgloname    \showgloname{<label>}

```
4552 \newcommand*{\showgloname}[1]{%
4553   \expandafter\show\csname glo@#1@name\endcsname
4554 }
```

\showglodesc    \showglodesc{<label>}

```
4555 \newcommand*{\showglodesc}[1]{%
4556   \expandafter\show\csname glo@#1@desc\endcsname
4557 }
```

\showglodescplural    \showglodescplural{<label>}

```
4558 \newcommand*{\showglodescplural}[1]{%
4559   \expandafter\show\csname glo@#1@descplural\endcsname
4560 }
```

\showglosort    \showglosort{<label>}

```
4561 \newcommand*{\showglosort}[1]{%
4562   \expandafter\show\csname glo@#1@sort\endcsname
4563 }
```

`\showglosymbol`    `\showglosymbol{\langle label \rangle}`

```
4564 \newcommand*{\showglosymbol}[1]{%
4565   \expandafter\show\csname glo@#1@symbol\endcsname
4566 }
```

`\showglosymbolplural`    `\showglosymbolplural{\langle label \rangle}`

```
4567 \newcommand*{\showglosymbolplural}[1]{%
4568   \expandafter\show\csname glo@#1@symbolplural\endcsname
4569 }
```

`\showgloindex`    `\showgloindex{\langle label \rangle}`

```
4570 \newcommand*{\showgloindex}[1]{%
4571   \expandafter\show\csname glo@#1@index\endcsname
4572 }
```

`\showgloflag`    `\showgloflag{\langle label \rangle}`

```
4573 \newcommand*{\showgloflag}[1]{%
4574   \expandafter\show\csname ifglo@#1@flag\endcsname
4575 }
```

`\showacronymlists`    `\showacronymlists`

Show list of glossaries that have been flagged as a list of acronyms.

```
4576 \newcommand*{\showacronymlists}{%
4577   \show\@glsacronymlists
4578 }
```

`\showglossaries`    `\showglossaries`

Show list of defined glossaries.

```
4579 \newcommand*{\showglossaries}{%
4580   \show\@glo@types
4581 }
```

`\showglossaryin`    `\showglossaryin{\langle glossary-label \rangle}`



Show the ‘in’ extension for the given glossary.

```
4582 \newcommand*{\showglossaryin}[1]{%
4583   \expandafter\show\csname @glotype@#1@in\endcsname
4584 }
```

`\showglossaryout`    `\showglossaryout{<glossary-label>}`

Show the ‘out’ extension for the given glossary.

```
4585 \newcommand*{\showglossaryout}[1]{%
4586   \expandafter\show\csname @glotype@#1@out\endcsname
4587 }
```

`\showglossarytitle`    `\showglossarytitle{<glossary-label>}`

Show the title for the given glossary.

```
4588 \newcommand*{\showglossarytitle}[1]{%
4589   \expandafter\show\csname @glotype@#1@title\endcsname
4590 }
```

`\showglossarycounter`    `\showglossarycounter{<glossary-label>}`

Show the counter for the given glossary.

```
4591 \newcommand*{\showglossarycounter}[1]{%
4592   \expandafter\show\csname @glotype@#1@counter\endcsname
4593 }
```

`\showglossaryentries`    `\showglossaryentries{<glossary-label>}`

Show the list of entry labels for the given glossary.

```
4594 \newcommand*{\showglossaryentries}[1]{%
4595   \expandafter\show\csname glolist@#1\endcsname
4596 }
```

## 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.

```
4597 \csname ifglsc-compatible-2.07\endcsname
4598   \RequirePackage{glossaries-compatible-207}
4599 \fi
```

## 2 Mfirstuc Documented Code

```
4600 \NeedsTeXFormat{LaTeX2e}
4601 \ProvidesPackage{mfirstuc}[2012/05/21 v1.06 (NLCT)]
```

Requires etoolbox:

```
4602 \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`.

```
4603 \newif\if@glscs
4604 \newtoks\@glsmfirst
4605 \newtoks\@glsmrest
4606 \def\makefirstuc#1{%
4607   \def\gls@argi{#1}%
4608   \ifx\gls@argi\@empty
      If the argument is empty, do nothing.
4609   \else

4610     \def\@gls@tmp{\ #1}%
4611     \@onelevel@sanitize\@gls@tmp
4612     \expandafter\@gls@checkcs\@gls@tmp\relax\relax
4613     \if@glscs
4614       \@gls@getbody #1{}\@nil
4615       \ifx\@gls@rest\@empty
4616         \glsmakefirstuc{#1}%
4617       \else
4618         \expandafter\@gls@split\@gls@rest\@nil
4619         \ifx\@gls@first\@empty
4620           \glsmakefirstuc{#1}%
4621         \else
4622           \expandafter\@glsmfirst\expandafter{\@gls@first}%
```

```

4623         \expandafter\@glsmrest\expandafter{\@gls@rest}%
4624         \edef\@gls@domfirstuc{\noexpand\@gls@body
4625         {\noexpand\glsmakefirstuc\the\@glsmfirst}%
4626         \the\@glsmrest}%
4627         \@gls@domfirstuc
4628     \fi
4629 \fi
4630 \else
4631     \glsmakefirstuc{#1}%
4632 \fi
4633 \fi
4634 }

```

Put first argument in \@gls@first and second argument in \@gls@rest:

```

4635 \def\@gls@split#1#2\@nil{%
4636 \def\@gls@first{#1}\def\@gls@rest{#2}%
4637 }

4638 \def\@gls@checkcs#1 #2#3\relax{%
4639 \def\@gls@argi{#1}\def\@gls@argii{#2}%
4640 \ifx\@gls@argi\@gls@argii
4641 \glscstrue
4642 \else
4643 \glscsfalse
4644 \fi
4645 }

```

Make first thing upper case:

```

4646 \def\@gls@makefirstuc#1{\MakeUppercase #1}

```

\glsmakefirstuc Provide a user command to make it easier to customise.

```

4647 \newcommand*\glsmakefirstuc[1]{\@gls@makefirstuc{#1}}

```

Get the first grouped argument and stores in \@gls@body.

```

4648 \def\@gls@getbody#1#\def\@gls@body{#1}\@gls@gobbletonil}

```

Scoup up everything to \@nil and store in \@gls@rest:

```

4649 \def\@gls@gobbletonil#1\@nil{\def\@gls@rest{#1}}

```

\xmakefirstuc Expand argument once before applying \makefirstuc (added v1.01).

```

4650 \newcommand*\xmakefirstuc[1]{%
4651 \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).

```

4652 \newcommand*\capitalisewords[1]{%
4653 \def\gls@add@space{%
4654 \mfu@capitalisewords#1 \@nil\mfu@endcap
4655 %\gls@add@space\makefirstuc{##1}\def\gls@add@space{ }%
4656 }

```

```

4657 \def\mfu@capitalisewords#1 #2\mfu@endcap{%
4658   \def\mfu@cap@first{#1}%
4659   \def\mfu@cap@second{#2}%
4660   \gls@add@space
4661   \makefirstuc{#1}%
4662   \def\gls@add@space{ }%
4663   \ifx\mfu@cap@second\@nnil
4664     \let\next\mfu@cap\mfu@noop
4665   \else
4666     \let\next\mfu@cap\mfu@capitalisewords
4667   \fi
4668   \next\mfu@cap#2\mfu@endcap
4669 }
4670 \def\mfu@noop#1\mfu@endcap{}

```

\xcapitalisewords Short-cut command:

```

4671 \newcommand*\xcapitalisewords[1]{%
4672   \expandafter\capitalisewords\expandafter{#1}%
4673 }

```

## 3 Glossary Styles

### 3.1 Glossary hyper-navigation definitions (glossary-hypernav package)

Package Definition:

```

4674 \ProvidesPackage{glossary-hypernav}[2007/07/04 v1.01 (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

```

4675 \newcommand*\glsnavhyperlink[3][\@glo@type]{%
4676   \edef\gls@grplabel{#2}\protected@edef\gls@grptitle{#3}%
4677   \@glslink{glsn:#1@#2}{#3}}

```

\glsnavhypertarget[⟨type⟩]{⟨label⟩}{⟨text⟩}

This command makes ⟨text⟩ a hypertext for the glossary group whose label is given by ⟨label⟩ in the glossary given by ⟨type⟩. If ⟨type⟩ is omitted, \@glo@type is used which is set by \printglossary to the current glossary label.

`\glsnavhypertarget`

```
4678 \newcommand*{\glsnavhypertarget}[3][\@glo@type]{%
```

Add this group to the aux file for re-run check.

```
4679 \protected@write\@auxout{}\string\@gls@hypergroup{#1}{#2}}%
```

Add the target.

```
4680 \gls@target{glsn:#1@#2}{#3}%
```

Check list of know groups to determine if a re-run is required.

```
4681 \expandafter\let
```

```
4682 \expandafter\@gls@list\csname @gls@hypergroup@list@#1\endcsname
```

Iterate through list and terminate loop if this group is found.

```
4683 \@for\@gls@elem:=\@gls@list\do{%
```

```
4684 \ifthenelse{\equal{\@gls@elem}{#2}}{\@endfortrue}{}}%
```

Check if list terminated prematurely.

```
4685 \if@endfor
```

```
4686 \else
```

This group was not included in the list, so issue a warning.

```
4687 \GlossariesWarningNoLine{Navigation panel
```

```
4688 for glossary type ‘#1’^^Jmissing group ‘#2’}%
```

```
4689 \gdef\gls@hypergroup@rerun{%
```

```
4690 \GlossariesWarningNoLine{Navigation panel
```

```
4691 has changed. Rerun LaTeX}}%
```

```
4692 \fi
```

```
4693 }
```

`\gls@hypergroup@rerun` Give a warning at the end if re-run required

```
4694 \let\gls@hypergroup@rerun\relax
```

```
4695 \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.

```
4696 \newcommand*{\@gls@hypergroup}[2]{%
```

```
4697 \@ifundefined{\@gls@hypergroup@list@#1}{%
```

```
4698 \expandafter\xdef\csname @gls@hypergroup@list@#1\endcsname{#2}}%
```

```
4699 }{%
```

```
4700 \expandafter\let\expandafter\@gls@tmp
```

```
4701 \csname @gls@hypergroup@list@#1\endcsname
```

```
4702 \expandafter\xdef\csname @gls@hypergroup@list@#1\endcsname{%
```

```
4703 \@gls@tmp,#2}}%
```

```
4704 }%
```

```
4705 }
```

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`

```
4706 \newcommand*\glsnavigation{%
4707 \def\@gls@between{%
4708 \@ifundefined{\@gls@hypergroup@list@\@glo@type}{%
4709   \def\@gls@list{%
4710 }{%
4711   \expandafter\let\expandafter\@gls@list
4712     \csname \@gls@hypergroup@list@\@glo@type\endcsname
4713 }%
4714 \@for\@gls@tmp:=\@gls@list\do{%
4715   \@gls@between
4716   \glsnavhyperlink{\@gls@tmp}{\glsgetgrouptitle{\@gls@tmp}}%
4717   \let\@gls@between\glshypernavsep%
4718 }%
4719 }
```

`\glshypernavsep` Separator for the hyper navigation bar.

```
4720 \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`

```
4721 \newcommand*\glssymbolnav{%
4722 \glsnavhyperlink{glssymbols}{\glsgetgrouptitle{glssymbols}}%
4723 \glshypernavsep
4724 \glsnavhyperlink{glsnumbers}{\glsgetgrouptitle{glsnumbers}}%
4725 \glshypernavsep
4726 }
```

### 3.2 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.

```
4727 \ProvidesPackage{glossary-inline}[2012/09/21 v3.03 (NLCT)]
```

`inline` Define the inline style.

```
4728 \newglossarystyle{inline}{%
```

Start of glossary sets up first empty separator between entries. (This is then changed by `\glossaryentryfield`)

```
4729 \renewenvironment{theglossary}%
4730 {%
4731     \def\gls@inlinesep{}%
4732     \def\gls@inlinesubsep{}%
4733     \def\gls@inlinepostchild{}%
4734 }%
4735 {\glspostinline}%
```

No header:

```
4736 \renewcommand*{\glossaryheader}{}%
```

No group headings (if heading is required, add `\glsinlinedopostchild` to start definition in case heading follows a child entry):

```
4737 \renewcommand*{\glsgroupheading}[1]{}%
```

Just display separator followed by name and description:

```
4738 \renewcommand{\glossaryentryfield}[5]{%
4739     \glsinlinedopostchild
4740     \gls@inlinesep
4741     \def\glo@desc{##3}%
4742     \def\@no@post@desc{\nopostdesc}%
4743     \glsentryitem{##1}\glsinlinenameformat{##1}{##2}%
4744     \ifx\glo@desc\@no@post@desc
4745         \glsinlineemptydescformat{##4}{##5}%
4746     \else
4747         \ifstrempy{##3}%
4748             {\glsinlineemptydescformat{##4}{##5}}%
4749             {\glsinlinedescformat{##3}{##4}{##5}}%
4750     \fi
4751     \ifglshaschildren{##1}%
4752     {%
4753         \glsresetsubentrycounter
4754         \glsinlineparentchildseparator
4755         \def\gls@inlinesubsep{}%
4756         \def\gls@inlinepostchild{\glsinlinepostchild}%
4757     }%
4758     {}%
4759     \def\gls@inlinesep{\glsinlineseparator}%
4760 }%
```

Sub-entries display description:

```
4761 \renewcommand{\glossarysubentryfield}[6]{%
4762     \gls@inlinesubsep%
4763     \glsinlinesubnameformat{##2}{##3}%
4764     \glssubentryitem{##2}\glsinlinesubdescformat{##4}{##5}{##6}%
4765     \def\gls@inlinesubsep{\glsinlinesubseparator}%
4766 }%
```

Nothing special between groups:

```

4767 \renewcommand*{\glsgroupskip}{}%
4768 }

\glsinlinedopostchild
4769 \newcommand*{\glsinlinedopostchild}{%
4770 \gls@inlinepostchild
4771 \def\gls@inlinepostchild{}%
4772 }

\glsinlineseparator Separator to use between entries.
4773 \newcommand*{\glsinlineseparator}{;\space}

\glsinlinesubseparator Separator to use between sub-entries.
4774 \newcommand*{\glsinlinesubseparator}{,\space}

\glsparentchildseparator Separator to use between parent and children.
4775 \newcommand*{\glsinlineparentchildseparator}{:\space}

\glsinlinepostchild Hook to use between child and next entry
4776 \newcommand*{\glsinlinepostchild}{}

\glspostinline Terminator for inline glossary.
4777 \newcommand*{\glspostinline}{\glspostdescription\space}

\glsinlinenameformat Formats the name of the entry (first argument label, second argument name):
4778 \newcommand*{\glsinlinenameformat}[2]{\glstarget{#1}{#2}}

\glsinlinedescformat Formats the entry's description, symbol and location list:
4779 \newcommand*{\glsinlinedescformat}[3]{\space#1}

\glsinlineemptydescformat Formats the entry's symbol and location list when the description is empty:
4780 \newcommand*{\glsinlineemptydescformat}[2]{\space#1}

\glsinlinesubnameformat Formats the name of the subentry (first argument label, second argument
name):
4781 \newcommand*{\glsinlinesubnameformat}[2]{\glstarget{#1}{#2}}

\glsinlinesubdescformat Formats the subentry's description, symbol and location list:
4782 \newcommand*{\glsinlinesubdescformat}[3]{\space#1}

```

### 3.3 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.

```
4783 \ProvidesPackage{glossary-list}[2012/11/11 v3.04 (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.

```
4784 \newglossarystyle{list}{%
```

Use description environment:

```
4785 \renewenvironment{theglossary}{%
```

```
4786 {\begin{description}}{\end{description}}%
```

No header at the start of the environment:

```
4787 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
4788 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries start a new item in the list:

```
4789 \renewcommand*{\glossaryentryfield}[5]{%
```

```
4790 \item[\glsentryitem{##1}\glstarget{##1}{##2}]
```

```
4791 ##3\glspostdescription\space ##5}%
```

Sub-entries continue on the same line:

```
4792 \renewcommand*{\glossarysubentryfield}[6]{%
```

```
4793 \glssubentryitem{##2}%
```

```
4794 \glstarget{##2}{\strut}##4\glspostdescription\space ##6.}%
```

```
4795 % \end{macrocode}
```

```
4796 % Add vertical space between groups:
```

```
4797 %\changes{3.03}{2012/09/21}{added check for glsnogroupskip}
```

```
4798 % \begin{macrocode}
```

```
4799 \renewcommand*{\glsgroupskip}{\ifglsgroupskip\else\indexspace\fi}%
```

```
4800 }
```

**listgroup** The listgroup style is like the list style, but the glossary groups have headings.

```
4801 \newglossarystyle{listgroup}{%
```

Base it on the list style:

```
4802 \glossarystyle{list}%
```

Each group has a heading:

```
4803 \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.

```
4804 \newglossarystyle{listhypergroup}{%
```

Base it on the list style:

```
4805 \glossarystyle{list}%
```

Add navigation links at the start of the environment:

```
4806 \renewcommand*{\glossaryheader}{%  
4807 \item[\glsnavigation]}%
```

Each group has a heading with a hypertarget:

```
4808 \renewcommand*{\glsgroupheading}[1]{%  
4809 \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.

```
4810 \newglossarystyle{altlist}{%
```

Base it on the list style:

```
4811 \glossarystyle{list}%
```

Main (level 0) entries start a new item in the list with a line break after the entry name:

```
4812 \renewcommand*{\glossaryentryfield}[5]{%  
4813 \item[\glsentryitem{##1}{\glstarget{##1}{##2}}]%
```

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.

```
4814 \mbox{}\par\nobreak\@afterheading  
4815 ##3\glspostdescription\space ##5}%
```

Sub-entries start a new paragraph:

```
4816 \renewcommand{\glossarysubentryfield}[6]{%  
4817 \par  
4818 \glssubentryitem{##2}%  
4819 \glstarget{##2}{\strut}##4\glspostdescription\space ##6}%  
4820 }
```

**altlistgroup** The altlistgroup glossary style is like the altlist style, but the glossary groups have headings.

```
4821 \newglossarystyle{altlistgroup}{%
```

Base it on the altlist style:

```
4822 \glossarystyle{altlist}%
```

Each group has a heading:

```
4823 \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.

```
4824 \newglossarystyle{altlisthypergroup}{%
```

Base it on the altlist style:

```
4825 \glossarystyle{altlist}%
```

Add navigation links at the start of the environment:

```
4826 \renewcommand*{\glossaryheader}{%
4827 \item[\glsnavigation]}%
```

Each group has a heading with a hypertarget:

```
4828 \renewcommand*{\glsgroupheading}[1]{%
4829 \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.

```
4830 \newglossarystyle{listdotted}{%
```

Base it on the list style:

```
4831 \glossarystyle{list}%
```

Each main (level 0) entry starts a new item:

```
4832 \renewcommand*{\glossaryentryfield}[5]{%
4833 \item[]\makebox[\glslistdottedwidth][l]{%
4834 \glssentryitem{##1}\glstarget{##1}{##2}%
4835 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##3}%
```

Sub entries have the same format as main entries:

```
4836 \renewcommand*{\glossarysubentryfield}[6]{%
4837 \item[]\makebox[\glslistdottedwidth][l]{%
4838 \glssubentryitem{##2}%
4839 \glstarget{##2}{##3}%
4840 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##4}%
4841 }
```

`\glslistdottedwidth`

```
4842 \newlength\glslistdottedwidth
4843 \setlength{\glslistdottedwidth}{.5\hsize}
```

**sublistdotted** This style is similar to the `glostylelistdotted` style, except that the main entries just have the name displayed.

```
4844 \newglossarystyle{sublistdotted}{%
```

Base it on the listdotted style:

```
4845 \glossarystyle{listdotted}%
```

Main (level 0) entries just display the name:

```
4846 \renewcommand*{\glossaryentryfield}[5]{%
4847 \item[\glssentryitem{##1}\glstarget{##1}{##2}]}%
4848 }
```

### 3.4 Glossary Styles using longtable (the glossary-long package)

The glossary styles defined in the package used the longtable environment in the glossary.

```
4849 \ProvidesPackage{glossary-long}[2012/09/21 v3.03 (NLCT)]
```

Requires the package:

```
4850 \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`.)

```
4851 \@ifundefined{glsdescwidth}{%
4852   \newlength{glsdescwidth
4853   \setlength{glsdescwidth}{0.6\hsize}
4854 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column.

```
4855 \@ifundefined{glspagelistwidth}{%
4856   \newlength{glspagelistwidth
4857   \setlength{glspagelistwidth}{0.1\hsize}
4858 }{}
```

`long` The long glossary style command which uses the longtable environment:

```
4859 \newglossarystyle{long}{%
```

Use longtable with two columns:

```
4860   \renewenvironment{theglossary}{%
4861     {\begin{longtable}{lp{glsdescwidth}}}%
4862     {\end{longtable}}%
```

Do nothing at the start of the environment:

```
4863   \renewcommand*{\glossaryheader}{}%
```

No heading between groups:

```
4864   \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries displayed in a row:

```
4865   \renewcommand*{\glossaryentryfield}[5]{%
4866     \glstryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\}%
```

Sub entries displayed on the following row without the name:

```
4867   \renewcommand*{\glossarysubentryfield}[6]{%
4868     &
4869     \glssubentryitem{##2}%
4870     \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\}%
```

Blank row between groups:

```
4871   \renewcommand*{\glsgroupskip}{\ifglsgnpgroupskip\else & \\fi}%
4872 }
```

**longborder** The longborder style is like the above, but with horizontal and vertical lines:

```
4873 \newglossarystyle{longborder}{%
```

Base it on the glosstylelong style:

```
4874 \glossarystyle{long}%
```

Use longtable with two columns with vertical lines between each column:

```
4875 \renewenvironment{theglossary}{%
```

```
4876 \begin{longtable}{|l|p{\glstdescwidth}|}{\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
4877 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
```

```
4878 }
```

**longheader** The longheader style is like the long style but with a header:

```
4879 \newglossarystyle{longheader}{%
```

Base it on the glosstylelong style:

```
4880 \glossarystyle{long}%
```

Set the table's header:

```
4881 \renewcommand*{\glossaryheader}{%
```

```
4882 \bfseries \entryname & \bfseries \descriptionname\\\endhead}%
```

```
4883 }
```

**longheaderborder** The longheaderborder style is like the long style but with a header and border:

```
4884 \newglossarystyle{longheaderborder}{%
```

Base it on the glosstylelongborder style:

```
4885 \glossarystyle{longborder}%
```

Set the table's header and add horizontal line to table's foot:

```
4886 \renewcommand*{\glossaryheader}{%
```

```
4887 \hline\bfseries \entryname & \bfseries \descriptionname\\\hline
```

```
4888 \endhead
```

```
4889 \hline\endfoot}%
```

```
4890 }
```

**long3col** The long3col style is like long but with 3 columns

```
4891 \newglossarystyle{long3col}{%
```

Use a longtable with 3 columns:

```
4892 \renewenvironment{theglossary}{%
```

```
4893 {\begin{longtable}{lp{\glstdescwidth}p{\glspagelistwidth}}}%
```

```
4894 {\end{longtable}}%
```

No table header:

```
4895 \renewcommand*{\glossaryheader}{}%
```

No headings between groups:

```
4896 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
4897 \renewcommand*{\glossaryentryfield}[5]{%
4898   \glstentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\\}%
```

Sub-entries on a separate row (no name, description in second column, page list in third column):

```
4899 \renewcommand*{\glossarysubentryfield}[6]{%
4900   &
4901   \glssubentryitem{##2}%
4902   \glstarget{##2}{\strut}##4 & ##6\\}%
```

Blank row between groups:

```
4903 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & &\\fi}%
4904 }
```

**long3colborder** The long3colborder style is like the long3col style but with a border:

```
4905 \newglossarystyle{long3colborder}{%
```

Base it on the glostylelong3col style:

```
4906 \glossarystyle{long3col}%
```

Use a longtable with 3 columns with vertical lines around them:

```
4907 \renewenvironment{theglossary}%
4908   {\begin{longtable}{|l|p{\glsdescwidth}|p{\glspagelistwidth}|}}%
4909   {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
4910 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
4911 }
```

**long3colheader** The long3colheader style is like long3col but with a header row:

```
4912 \newglossarystyle{long3colheader}{%
```

Base it on the glostylelong3col style:

```
4913 \glossarystyle{long3col}%
```

Set the table's header:

```
4914 \renewcommand*{\glossaryheader}{%
4915   \bfseries\entryname&\bfseries\descriptionname&
4916   \bfseries\pagelistname\\endhead}%
4917 }
```

**long3colheaderborder** The long3colheaderborder style is like the above but with a border

```
4918 \newglossarystyle{long3colheaderborder}{%
```

Base it on the glostylelong3colborder style:

```
4919 \glossarystyle{long3colborder}%
```

Set the table's header and add horizontal line at table's foot:

```
4920 \renewcommand*{\glossaryheader}{%
4921 \hline
4922 \bfseries\entryname&\bfseries\descriptionname&
4923 \bfseries\pagelistname\\\hline\endhead
4924 \hline\endfoot}%
4925 }
```

**long4col** The long4col style has four columns where the third column contains the value of the associated symbol key.

```
4926 \newglossarystyle{long4col}{%
```

Use a longtable with 4 columns:

```
4927 \renewenvironment{theglossary}%
4928 {\begin{longtable}{llll}}%
4929 {\end{longtable}}%
```

No table header:

```
4930 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
4931 \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):

```
4932 \renewcommand*{\glossaryentryfield}[5]{%
4933 \glstentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\\}%
```

Sub entries on a single row with no name (description in second column, symbol in third column, page list in last column):

```
4934 \renewcommand*{\glossarysubentryfield}[6]{%
4935 &
4936 \glssubentryitem{##2}%
4937 \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%
```

Blank row between groups:

```
4938 \renewcommand*{\glsgroupskip}{\ifglsgnoglobskip\else & & & \\fi}%
4939 }
```

**long4colheader** The long4colheader style is like long4col but with a header row.

```
4940 \newglossarystyle{long4colheader}{%
```

Base it on the glostylelong4col style:

```
4941 \glossarystyle{long4col}%
```

Table has a header:

```
4942 \renewcommand*{\glossaryheader}{%
4943 \bfseries\entryname&\bfseries\descriptionname&
4944 \bfseries \symbolname&
4945 \bfseries\pagelistname\\\endhead}%
4946 }
```

`long4colborder` The `long4colborder` style is like `long4col` but with a border.

```
4947 \newglossarystyle{long4colborder}{%
```

Base it on the `glostylelong4col` style:

```
4948 \glossarystyle{long4col}%
```

Use a longtable with 4 columns surrounded by vertical lines:

```
4949 \renewenvironment{theglossary}%
```

```
4950 {\begin{longtable}{|l|l|l|l|}}%
```

```
4951 {\end{longtable}}%
```

Add horizontal lines to the head and foot of the table:

```
4952 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
```

```
4953 }
```

`long4colheaderborder` The `long4colheaderborder` style is like the above but with a border.

```
4954 \newglossarystyle{long4colheaderborder}{%
```

Base it on the `glostylelong4col` style:

```
4955 \glossarystyle{long4col}%
```

Use a longtable with 4 columns surrounded by vertical lines:

```
4956 \renewenvironment{theglossary}%
```

```
4957 {\begin{longtable}{|l|l|l|l|}}%
```

```
4958 {\end{longtable}}%
```

Add table header and horizontal line at the table's foot:

```
4959 \renewcommand*{\glossaryheader}{%
```

```
4960 \hline\bfseries\entryname&\bfseries\descriptionname&
```

```
4961 \bfseries \symbolname&
```

```
4962 \bfseries\pagelistname\\\hline\endhead\hline\endfoot}%
```

```
4963 }
```

`altlong4col` The `altlong4col` style is like the `long4col` style but can have multiline descriptions and page lists.

```
4964 \newglossarystyle{altlong4col}{%
```

Base it on the `glostylelong4col` style:

```
4965 \glossarystyle{long4col}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
4966 \renewenvironment{theglossary}%
```

```
4967 {\begin{longtable}{lp{\glsgdescwidth}lp{\glspagelistwidth}}%
```

```
4968 {\end{longtable}}%
```

```
4969 }
```

`altlong4colheader` The `altlong4colheader` style is like `altlong4col` but with a header row.

```
4970 \newglossarystyle{altlong4colheader}{%
```

Base it on the `glostylelong4colheader` style:

```
4971 \glossarystyle{long4colheader}%
```



Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
4972 \renewenvironment{theglossary}%
4973   {\begin{longtable}{lp{\glsdescwidth}lp{\glspagelistwidth}}}%
4974   {\end{longtable}}}%
4975 }
```

**altlong4colborder** The altlong4colborder style is like altlong4col but with a border.

```
4976 \newglossarystyle{altlong4colborder}{%
```

Base it on the glostylelong4colborder style:

```
4977 \glossarystyle{long4colborder}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
4978 \renewenvironment{theglossary}%
4979   {\begin{longtable}{|lp{\glsdescwidth}|lp{\glspagelistwidth}|}%
4980   {\end{longtable}}}%
4981 }
```

**ong4colheaderborder** The altlong4colheaderborder style is like the above but with a header as well as a border.

```
4982 \newglossarystyle{altlong4colheaderborder}{%
```

Base it on the glostylelong4colheaderborder style:

```
4983 \glossarystyle{long4colheaderborder}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
4984 \renewenvironment{theglossary}%
4985   {\begin{longtable}{|lp{\glsdescwidth}|lp{\glspagelistwidth}|}%
4986   {\end{longtable}}}%
4987 }
```

### 3.5 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.

```
4988 \ProvidesPackage{glossary-longragged}[2012/09/21 v3.03 (NLCT)]
```

Requires the package:

```
4989 \RequirePackage{array}
```

Requires the package:

```
4990 \RequirePackage{longtable}
```

**\glsdescwidth** This is a length that governs the width of the description column. This may have already been defined.

```

4991 \@ifundefined{glsdescwidth}{%
4992   \newlength{glsdescwidth
4993   \setlength{glsdescwidth}{0.6\hsize}
4994 }{}

```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

```

4995 \@ifundefined{glspagelistwidth}{%
4996   \newlength{glspagelistwidth
4997   \setlength{glspagelistwidth}{0.1\hsize}
4998 }{}

```

`longragged` The longragged glossary style is like the long but uses ragged right formatting for the description column.

```

4999 \newglossarystyle{longragged}{%

```

Use longtable with two columns:

```

5000   \renewenvironment{theglossary}%
5001     {\begin{longtable}[1>{\raggedright}p{glsdescwidth}}}%
5002     {\end{longtable}}%

```

Do nothing at the start of the environment:

```

5003   \renewcommand*{\glossaryheader}{}%

```

No heading between groups:

```

5004   \renewcommand*{\glsgroupheading}[1]{}%

```

Main (level 0) entries displayed in a row:

```

5005   \renewcommand*{\glossaryentryfield}[5]{%
5006     \glstentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%
5007     \tabularnewline}%

```

Sub entries displayed on the following row without the name:

```

5008   \renewcommand*{\glossarysubentryfield}[6]{%
5009     &
5010     \glssubentryitem{##2}%
5011     \glstarget{##2}{\strut}##4\glspostdescription\space ##6%
5012     \tabularnewline}%

```

Blank row between groups:

```

5013   \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & \tabularnewline\fi}%
5014 }

```

`longraggedborder` The longraggedborder style is like the above, but with horizontal and vertical lines:

```

5015 \newglossarystyle{longraggedborder}{%

```

Base it on the `glostylelongragged` style:

```

5016   \glossarystyle{longragged}%

```

Use longtable with two columns with vertical lines between each column:

```
5017 \renewenvironment{theglossary}{%
5018 \begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|}%
5019 {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
5020 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
5021 }
```

**longraggedheader** The longraggedheader style is like the longragged style but with a header:

```
5022 \newglossarystyle{longraggedheader}{%
```

Base it on the glostylelongragged style:

```
5023 \glossarystyle{longragged}%
```

Set the table's header:

```
5024 \renewcommand*{\glossaryheader}{%
5025 \bfseries \entryname & \bfseries \descriptionname
5026 \tabularnewline\endhead}%
5027 }
```

**longraggedheaderborder** The longraggedheaderborder style is like the longragged style but with a header and border:

```
5028 \newglossarystyle{longraggedheaderborder}{%
```

Base it on the glostylelongraggedborder style:

```
5029 \glossarystyle{longraggedborder}%
```

Set the table's header and add horizontal line to table's foot:

```
5030 \renewcommand*{\glossaryheader}{%
5031 \hline\bfseries \entryname & \bfseries \descriptionname
5032 \tabularnewline\hline
5033 \endhead
5034 \hline\endfoot}%
5035 }
```

**longragged3col** The longragged3col style is like longragged but with 3 columns

```
5036 \newglossarystyle{longragged3col}{%
```

Use a longtable with 3 columns:

```
5037 \renewenvironment{theglossary}{%
5038 {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}%
5039 >{\raggedright}p{\gls pagelistwidth}}}%
5040 {\end{longtable}}%
```

No table header:

```
5041 \renewcommand*{\glossaryheader}{}%
```

No headings between groups:

```
5042 \renewcommand*{\gls groupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
5043 \renewcommand*{\glossaryentryfield}[5]{%
5044   \glstentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\tabularnewline}%
```

Sub-entries on a separate row (no name, description in second column, page list in third column):

```
5045 \renewcommand*{\glossarysubentryfield}[6]{%
5046   &
5047   \glssubentryitem{##2}%
5048   \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%
```

Blank row between groups:

```
5049 \renewcommand*{\glsnoglobskip}{\ifglsnogroupskip\else & &\tabularnewline\fi}%
5050 }
```

`longragged3colborder` The `longragged3colborder` style is like the `longragged3col` style but with a border:

```
5051 \newglossarystyle{longragged3colborder}{%
```

Base it on the `glostylelongragged3col` style:

```
5052 \glossarystyle{longragged3col}%
```

Use a `longtable` with 3 columns with vertical lines around them:

```
5053 \renewenvironment{theglossary}%
5054   {\begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|%
5055    >{\raggedright}p{\glspagelistwidth}|}%
5056   {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
5057 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
5058 }
```

`longragged3colheader` The `longragged3colheader` style is like `longragged3col` but with a header row:

```
5059 \newglossarystyle{longragged3colheader}{%
```

Base it on the `glostylelongragged3col` style:

```
5060 \glossarystyle{longragged3col}%
```

Set the table's header:

```
5061 \renewcommand*{\glossaryheader}{%
5062   \bfseries\entryname&\bfseries\descriptionname&
5063   \bfseries\pagelistname\tabularnewline\endhead}%
5064 }
```

`longragged3colheaderborder` The `longragged3colheaderborder` style is like the above but with a border

```
5065 \newglossarystyle{longragged3colheaderborder}{%
```

Base it on the `glostylelongragged3colborder` style:

```
5066 \glossarystyle{longragged3colborder}%
```

Set the table's header and add horizontal line at table's foot:

```
5067 \renewcommand*{\glossaryheader}{%
5068 \hline
5069 \bfseries\entryname&\bfseries\descriptionname&
5070 \bfseries\pagelistname\tabularnewline\hline\endhead
5071 \hline\endfoot}%
5072 }
```

`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.

```
5073 \newglossarystyle{altlongragged4col}{%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
5074 \renewenvironment{theglossary}{%
5075 {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%
5076 >{\raggedright}p{\glspagelistwidth}}}%
5077 {\end{longtable}}}%
```

No table header:

```
5078 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
5079 \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):

```
5080 \renewcommand*{\glossaryentryfield}[5]{%
5081 \glstarget{##1}{\glstarget{##1}{##2} & ##3 & ##4 & ##5\tabularnewline}%
```

Sub entries on a single row with no name (description in second column, symbol in third column, page list in last column):

```
5082 \renewcommand*{\glossarysubentryfield}[6]{%
5083 &
5084 \glssubentryitem{##2}%
5085 \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%
```

Blank row between groups:

```
5086 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & & \tabularnewline\fi}%
5087 }
```

`ongragged4colheader` The `altlongragged4colheader` style is like `altlongragged4col` but with a header row.

```
5088 \newglossarystyle{altlongragged4colheader}{%
```

Base it on the `glostylealtlongragged4col` style:

```
5089 \glossarystyle{altlongragged4col}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
5090 \renewenvironment{theglossary}%
5091   {\begin{longtable}{l>{\raggedright}p{\glsgdescwidth}l%
5092     >{\raggedright}p{\glspagelistwidth}}}%
5093   {\end{longtable}}%
```

Table has a header:

```
5094 \renewcommand*{\glossaryheader}{%
5095   \bfseries\entryname&\bfseries\descriptionname&
5096   \bfseries \symbolname&
5097   \bfseries\pagelistname\tabularnewline\endhead}%
5098 }
```

`ongragged4colborder` The `altlongragged4colborder` style is like `altlongragged4col` but with a border.

```
5099 \newglossarystyle{altlongragged4colborder}{%
```

Base it on the `glostylealtlongragged4col` style:

```
5100 \glossarystyle{altlongragged4col}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
5101 \renewenvironment{theglossary}%
5102   {\begin{longtable}{|l|>{\raggedright}p{\glsgdescwidth}|l|}%
5103     >{\raggedright}p{\glspagelistwidth}|}%
5104   {\end{longtable}}%
```

Add horizontal lines to the head and foot of the table:

```
5105 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
5106 }
```

`ged4colheaderborder` The `altlongragged4colheaderborder` style is like the above but with a header as well as a border.

```
5107 \newglossarystyle{altlongragged4colheaderborder}{%
```

Base it on the `glostylealtlongragged4col` style:

```
5108 \glossarystyle{altlongragged4col}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
5109 \renewenvironment{theglossary}%
5110   {\begin{longtable}{|l|>{\raggedright}p{\glsgdescwidth}|l|}%
5111     >{\raggedright}p{\glspagelistwidth}|}%
5112   {\end{longtable}}%
```

Add table header and horizontal line at the table's foot:

```
5113 \renewcommand*{\glossaryheader}{%
5114   \hline\bfseries\entryname&\bfseries\descriptionname&
5115   \bfseries \symbolname&
5116   \bfseries\pagelistname\tabularnewline\hline\endhead
5117   \hline\endfoot}%
5118 }
```

### 3.6 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.

```
5119 \ProvidesPackage{glossary-mcols}[2012/05/21 v1.0 (NLCT)]
```

Required packages:

```
5120 \RequirePackage{multicol}
```

```
5121 \RequirePackage{glossary-tree}
```

`\glsmcols` Define macro in which to store the number of columns. (Defaults to 2.)

```
5122 \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 multicol, but the title isn't part of the glossary style.)

```
5123 \newglossarystyle{mcolindex}{%
5124   \glossarystyle{index}%
5125   \renewenvironment{theglossary}%
5126     {%
5127       \begin{multicols}{2}
5128       \setlength{\parindent}{0pt}%
5129       \setlength{\parskip}{0pt plus 0.3pt}%
5130       \let\item\@idxitem}%
5131     {\end{multicols}}%
5132 }
```

`mcolindexgroup` As `mcolindex` but has headings:

```
5133 \newglossarystyle{mcolindexgroup}{%
5134   \glossarystyle{mcolindex}%
5135   \renewcommand*\glsgroupheading[1]{%
5136     \item\textbf{\glsgroupheading{##1}}\indexspace}%
5137 }
```

`mcolindexhypergroup` The `mcolindexhypergroup` style is like the `mcolindexgroup` style but has hyper navigation.

```
5138 \newglossarystyle{mcolindexhypergroup}{%
```

Base it on the `glostylemcolindex` style:

```
5139   \glossarystyle{mcolindex}%
```

Put navigation links to the groups at the start of the glossary:

```
5140   \renewcommand*\glossaryheader{%
5141     \item\textbf{\glshnavigation}\indexspace}%
```

Add a heading for each group (with a target). The group's title is in bold followed by a vertical gap.

```
5142   \renewcommand*\glsgroupheading[1]{%
5143     \item\textbf{\glshnavhypertarget{##1}}{\glsgroupheading{##1}}}%
5144   \indexspace}%
5145 }
```

**mcoltree** Multi-column index style. Same as the tree, but puts the glossary in multiple columns.

```
5146 \newglossarystyle{mcoltree}{%
5147   \glossarystyle{tree}%
5148   \renewenvironment{theglossary}%
5149   {%
5150     \begin{multicols}{2}
5151     \setlength{\parindent}{0pt}%
5152     \setlength{\parskip}{0pt plus 0.3pt}%
5153   }%
5154   {\end{multicols}}%
5155 }
```

**mcoltreegroup** Like the mcoltree style but the glossary groups have headings.

```
5156 \newglossarystyle{mcoltreegroup}{%
  Base it on the glostylemcoltree style:
5157   \glossarystyle{mcoltree}%
  Each group has a heading (in bold) followed by a vertical gap):
5158   \renewcommand{\glsgroupheading}[1]{\par
5159     \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
5160 }
```

**mcoltreehypergroup** The mcoltreehypergroup style is like the treegroup style, but has a set of links to the groups at the start of the glossary.

```
5161 \newglossarystyle{mcoltreehypergroup}{%
  Base it on the glostylemcoltree style:
5162   \glossarystyle{mcoltree}%
  Put navigation links to the groups at the start of the theglossary environment:
5163   \renewcommand*{\glsgroupheading}{%
5164     \par\noindent\textbf{\glsnavigation}\par\indexspace}%
  Each group has a heading (in bold with a target) followed by a vertical gap):
5165   \renewcommand*{\glsgroupheading}[1]{%
5166     \par\noindent
5167     \textbf{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}\par
5168     \indexspace}%
5169 }
```

**mcoltreename** Multi-column index style. Same as the treename, but puts the glossary in multiple columns.

```
5170 \newglossarystyle{mcoltreename}{%
5171   \glossarystyle{treename}%
5172   \renewenvironment{theglossary}%
5173   {%
5174     \begin{multicols}{2}
5175     \setlength{\parindent}{0pt}%
5176   }%
5177   {\end{multicols}}%
```



```

5176     \setlength{\parskip}{0pt plus 0.3pt}%
5177   }%
5178   {\end{multicols}}}%
5179 }

```

**mcoltreenonamegroup** Like the mcoltreenoname style but the glossary groups have headings.

```

5180 \newglossarystyle{mcoltreenonamegroup}{%
    Base it on the glostylemcoltreenoname style:
5181   \glossarystyle{mcoltreenoname}%
    Give each group a heading:
5182   \renewcommand{\glsgroupheading}[1]{\par
5183     \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
5184 }

```

**treenonamehypergroup** The mcoltreenonamehypergroup style is like the mcoltreenonamegroup style, but has a set of links to the groups at the start of the glossary.

```

5185 \newglossarystyle{mcoltreenonamehypergroup}{%
    Base it on the glostylemcoltreenoname style:
5186   \glossarystyle{mcoltreenoname}%
    Put navigation links to the groups at the start of the theglossary environment:
5187   \renewcommand*{\glossaryheader}{%
5188     \par\noindent\textbf{\glsnavigation}\par\indexspace}%
    Each group has a heading (in bold with a target) followed by a vertical gap):
5189   \renewcommand*{\glsgroupheading}[1]{%
5190     \par\noindent
5191     \textbf{\glsnavigationhypertarget{##1}{\glsgrouptitle{##1}}}\par
5192     \indexspace}%
5193 }

```

**mcolalmtree** Multi-column index style. Same as the almtree, but puts the glossary in multiple columns.

```

5194 \newglossarystyle{mcolalmtree}{%
5195   \glossarystyle{almtree}%
5196   \renewenvironment{theglossary}%
5197   {%
5198     \begin{multicols}{2}%
5199     \def\@gls@prevlevel{-1}%
5200     \mbox{}\par
5201   }%
5202   {\par\end{multicols}}%
5203 }

```

**mcolalmtreegroup** Like the mcolalmtree style but the glossary groups have headings.

```

5204 \newglossarystyle{mcolalmtreegroup}{%

```

Base it on the `glostylemcolalttree` style:

```
5205 \glossarystyle{mcolalttree}%
    Give each group a heading.
5206 \renewcommand{\glsgroupheading}[1]{\par
5207   \def\@gls@prevlevel{-1}%
5208   \hangindent0pt\relax
5209   \parindent0pt\relax
5210   \textbf{\glsgetgrouptitle{##1}}\par\indexspace}%
5211 }
```

`colalttreehypergroup` The `mcolalttreehypergroup` style is like the `mcolalttreegroup` style, but has a set of links to the groups at the start of the glossary.

```
5212 \newglossarystyle{mcolalttreehypergroup}{%
```

Base it on the `glostylemcolalttree` style:

```
5213 \glossarystyle{mcolalttree}%
    Put the navigation links in the header
5214 \renewcommand*{\glossaryheader}{%
5215   \par
5216   \def\@gls@prevlevel{-1}%
5217   \hangindent0pt\relax
5218   \parindent0pt\relax
5219   \textbf{\glsnavigation}\par\indexspace}%
    Put a hypertarget at the start of each group
```

```
5220 \renewcommand*{\glsgroupheading}[1]{%
5221   \par
5222   \def\@gls@prevlevel{-1}%
5223   \hangindent0pt\relax
5224   \parindent0pt\relax
5225   \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
5226   \indexspace}}
```

### 3.7 Glossary Styles using supertabular environment (glossary-super package)

The glossary styles defined in the package use the `supertabular` environment.

```
5227 \ProvidesPackage{glossary-super}[2012/09/21 v3.03 (NLCT)]
```

Requires the package:

```
5228 \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.

```
5229 \@ifundefined{glsdescwidth}{%
5230   \newlength\glsdescwidth
5231   \setlength{\glsdescwidth}{0.6\hsize}
5232 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined if has been loaded.

```
5233 \@ifundefined{glspagelistwidth}{%
5234   \newlength{glspagelistwidth}
5235   \setlength{glspagelistwidth}{0.1\hsize}
5236 }{}
```

**super** The super glossary style uses the supertabular environment (it uses lengths defined in the package.)

```
5237 \newglossarystyle{super}{%
```

Put the glossary in a supertabular environment with two columns and no head or tail:

```
5238   \renewenvironment{theglossary}%
5239     {\tablehead{}\tabletail}%
5240     \begin{supertabular}{lp{glstdescwidth}}%
5241     {\end{supertabular}}%
```

Do nothing at the start of the table:

```
5242   \renewcommand*{glossaryheader}{}%
```

No group headings:

```
5243   \renewcommand*{glsgroupheading}[1]{}%
```

Main (level 0) entries put in a row (name in first column, description and page list in second column):

```
5244   \renewcommand*{glossaryentryfield}[5]{%
5245     \glstentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\}%
```

Sub entries put in a row (no name, description and page list in second column):

```
5246   \renewcommand*{glossarysubentryfield}[6]{%
5247     &
5248     \glssubentryitem{##2}%
5249     \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\}%
```

Blank row between groups:

```
5250   \renewcommand*{glsgroupskip}{\ifglsnogroupskip\else & \\fi}%
5251 }
```

**superborder** The superborder style is like the above, but with horizontal and vertical lines:

```
5252 \newglossarystyle{superborder}{%
```

Base it on the glostylesuper style:

```
5253   \glossarystyle{super}%
```

Put the glossary in a supertabular environment with two columns and a horizontal line in the head and tail:

```
5254   \renewenvironment{theglossary}%
5255     {\tablehead{\hline}\tabletail{\hline}%
5256     \begin{supertabular}{|lp{glstdescwidth}|}%
5257     {\end{supertabular}}%
5258 }
```

**superheader** The superheader style is like the super style, but with a header:

```
5259 \newglossarystyle{superheader}{%
```

Base it on the glostylesuper style:

```
5260 \glossarystyle{super}{%
```

Put the glossary in a supertabular environment with two columns, a header and no tail:

```
5261 \renewenvironment{theglossary}{%
```

```
5262 {\tablehead{\bfseries \entryname & \bfseries \descriptionname\\}%
```

```
5263 \tabletail{}}%
```

```
5264 \begin{supertabular}{lp{\glsgdescwidth}}%
```

```
5265 {\end{supertabular}}%
```

```
5266 }
```

**superheaderborder** The superheaderborder style is like the super style but with a header and border:

```
5267 \newglossarystyle{superheaderborder}{%
```

Base it on the glostylesuper style:

```
5268 \glossarystyle{super}{%
```

Put the glossary in a supertabular environment with two columns, a header and horizontal lines above and below the table:

```
5269 \renewenvironment{theglossary}{%
```

```
5270 {\tablehead{\hline\bfseries \entryname &
```

```
5271 \bfseries \descriptionname\\hline}%
```

```
5272 \tabletail{\hline}
```

```
5273 \begin{supertabular}{|lp{\glsgdescwidth}|}%
```

```
5274 {\end{supertabular}}%
```

```
5275 }
```

**super3col** The super3col style is like the super style, but with 3 columns:

```
5276 \newglossarystyle{super3col}{%
```

Put the glossary in a supertabular environment with three columns and no head or tail:

```
5277 \renewenvironment{theglossary}{%
```

```
5278 {\tablehead{}\tabletail{}}%
```

```
5279 \begin{supertabular}{lp{\glsgdescwidth}p{\glspagelistwidth}}%
```

```
5280 {\end{supertabular}}%
```

Do nothing at the start of the table:

```
5281 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
5282 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
5283 \renewcommand*{\glossaryentryfield}[5]{}%
```

```
5284 \glstentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\\}%
```

Sub entries on a row (no name, description in second column, page list in last column):

```
5285 \renewcommand*{\glossarysubentryfield}[6]{%
5286     &
5287     \glssubentryitem{##2}%
5288     \glstarget{##2}{\strut}##4 & ##6\\}%
```

Blank row between groups:

```
5289 \renewcommand*{\glsnoglobskip}{\ifglsnogroupskip\else & &\\fi}%
5290 }
```

**super3colborder** The super3colborder style is like the super3col style, but with a border:

```
5291 \newglossarystyle{super3colborder}{%
    Base it on the glostylesuper3col style:
5292 \glossarystyle{super3col}%
    Put the glossary in a supertabular environment with three columns and a horizontal line in the head and tail:
5293 \renewenvironment{theglossary}%
5294     {\tablehead{\hline}\tabletail{\hline}%
5295     \begin{supertabular}{|l|p{\glsdescwidth}|p{\glspagelistwidth}|}%
5296     {\end{supertabular}}%
5297 }
```

**super3colheader** The super3colheader style is like the super3col style but with a header row:

```
5298 \newglossarystyle{super3colheader}{%
    Base it on the glostylesuper3col style:
5299 \glossarystyle{super3col}%
    Put the glossary in a supertabular environment with three columns, a header and no tail:
5300 \renewenvironment{theglossary}%
5301     {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
5302     \bfseries\pagelistname\\}\tabletail{}}%
5303     \begin{supertabular}{lp{\glsdescwidth}p{\glspagelistwidth}}}%
5304     {\end{supertabular}}%
5305 }
```

**super3colheaderborder** The super3colheaderborder style is like the super3col style but with a header and border:

```
5306 \newglossarystyle{super3colheaderborder}{%
    Base it on the glostylesuper3colborder style:
5307 \glossarystyle{super3colborder}%
    Put the glossary in a supertabular environment with three columns, a header with horizontal lines and a horizontal line in the tail:
5308 \renewenvironment{theglossary}%
5309     {\tablehead{\hline
```

```

5310      \bfseries\entryname&\bfseries\descriptionname&
5311      \bfseries\pagelistname\\\hline}%
5312      \tabletail{\hline}%
5313      \begin{supertabular}{|l|p{\glsgdescwidth}|p{\glspagelistwidth}|}%
5314      {\end{supertabular}}}%
5315 }

```

**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.

```
5316 \newglossarystyle{super4col}{%
```

Put the glossary in a supertabular environment with four columns and no head or tail:

```

5317   \renewenvironment{theglossary}%
5318   {\tablehead{}\tabletail}%
5319   \begin{supertabular}{|l|l|l|l|}%
5320   \end{supertabular}}%

```

Do nothing at the start of the table:

```
5321   \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
5322   \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:

```

5323   \renewcommand*{\glossaryentryfield}[5]{%
5324   \glstryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\\}%

```

Sub entries on a row with no name, the description in the second column, symbol in third column and page list in last column:

```

5325   \renewcommand*{\glossarysubentryfield}[6]{%
5326   &
5327   \glssubentryitem{##2}%
5328   \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%

```

Blank row between groups:

```

5329   \renewcommand*{\glsgroupskip}{\ifglsgnigroupskip\else & & \\\fi}%
5330 }

```

**super4colheader** The **super4colheader** style is like the **super4col** but with a header row.

```
5331 \newglossarystyle{super4colheader}{%
```

Base it on the **glostylesuper4col** style:

```
5332   \glossarystyle{super4col}%
```

Put the glossary in a supertabular environment with four columns, a header and no tail:

```

5333   \renewenvironment{theglossary}%
5334   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
5335   \bfseries\symbolname &

```

```

5336      \bfseries\pagelistname\\}%
5337      \tabletail{}}%
5338      \begin{supertabular}{|l|l|l|l|}%
5339      {\end{supertabular}}}%
5340 }

```

**super4colborder** The super4colborder style is like the super4col but with a border.

```
5341 \newglossarystyle{super4colborder}{%
```

Base it on the glostylesuper4col style:

```
5342 \glossarystyle{super4col}%
```

Put the glossary in a supertabular environment with four columns and a horizontal line in the head and tail:

```

5343 \renewenvironment{theglossary}%
5344 {\tablehead{\hline}\tabletail{\hline}%
5345 \begin{supertabular}{|l|l|l|l|}%
5346 {\end{supertabular}}}%
5347 }

```

**super4colheaderborder** The super4colheaderborder style is like the super4col but with a header and border.

```
5348 \newglossarystyle{super4colheaderborder}{%
```

Base it on the glostylesuper4col style:

```
5349 \glossarystyle{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:

```

5350 \renewenvironment{theglossary}%
5351 {\tablehead{\hline\bfseries\entryname&\bfseries\descriptionname&
5352 \bfseries\symbolname &
5353 \bfseries\pagelistname\\hline}\tabletail{\hline}%
5354 \begin{supertabular}{|l|l|l|l|}%
5355 {\end{supertabular}}}%
5356 }

```

**altsuper4col** The altsuper4col glossary style is like super4col but has provision for multiline descriptions.

```
5357 \newglossarystyle{altsuper4col}{%
```

Base it on the glostylesuper4col style:

```
5358 \glossarystyle{super4col}%
```

Put the glossary in a supertabular environment with four columns and no head or tail:

```

5359 \renewenvironment{theglossary}%
5360 {\tablehead{}\tabletail{}}%
5361 \begin{supertabular}{lp{\glstdescwidth}lp{\glspagelistwidth}}}%
5362 {\end{supertabular}}}%
5363 }

```

**altsuper4colheader** The altsuper4colheader style is like the altsuper4col but with a header row.

```

5364 \newglossarystyle{altsuper4colheader}{%
    Base it on the glostylesuper4colheader style:
5365   \glossarystyle{super4colheader}%

    Put the glossary in a supertabular environment with four columns, a header and
    no tail:
5366   \renewenvironment{theglossary}%
5367     {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
5368       \bfseries\symbolname &
5369       \bfseries\pagelistname\\}\tabletail{}}%
5370     \begin{supertabular}{lp{\glsgdescwidth}lp{\glspagelistwidth}}}%
5371     {\end{supertabular}}}%
5372 }
```

**altsuper4colborder** The altsuper4colborder style is like the altsuper4col but with a border.

```

5373 \newglossarystyle{altsuper4colborder}{%
    Base it on the glostylesuper4colborder style:
5374   \glossarystyle{super4colborder}%

    Put the glossary in a supertabular environment with four columns and a horizontal
    line in the head and tail:
5375   \renewenvironment{theglossary}%
5376     {\tablehead{\hline}\tabletail{\hline}%
5377     \begin{supertabular}%
5378       {\lllp{\glsgdescwidth}lllp{\glspagelistwidth}}}%
5379     {\end{supertabular}}}%
5380 }
```

**per4colheaderborder** The altsuper4colheaderborder style is like the altsuper4col but with a header and border.

```

5381 \newglossarystyle{altsuper4colheaderborder}{%
    Base it on the glostylesuper4colheaderborder style:
5382   \glossarystyle{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:
5383   \renewenvironment{theglossary}%
5384     {\tablehead{\hline
5385       \bfseries\entryname &
5386       \bfseries\descriptionname &
5387       \bfseries\symbolname &
5388       \bfseries\pagelistname\\\hline}%
5389     \tabletail{\hline}%
5390     \begin{supertabular}%
5391       {\lllp{\glsgdescwidth}lllp{\glspagelistwidth}}}%
5392     {\end{supertabular}}}%
5393 }
```



### 3.8 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.

5394 \ProvidesPackage{glossary-superragged}[2012/09/21 v3.03 (NLCT)]

Requires the package:

5395 \RequirePackage{array}

Requires the package:

5396 \RequirePackage{supertabular}

`\glsdescwidth` This is a length that governs the width of the description column. This may already have been defined.

5397 \@ifundefined{glsdescwidth}{%

5398 \newlength{glsdescwidth}

5399 \setlength{glsdescwidth}{0.6\hsize}

5400 }{}

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

5401 \@ifundefined{glspagelistwidth}{%

5402 \newlength{glspagelistwidth}

5403 \setlength{glspagelistwidth}{0.1\hsize}

5404 }{}

`superragged` The superragged glossary style uses the supertabular environment.

5405 \newglossarystyle{superragged}{%

Put the glossary in a supertabular environment with two columns and no head or tail:

5406 \renewenvironment{theglossary}{%

5407 {\tablehead{}\tabletail}{%

5408 \begin{supertabular}{1>\raggedright}p{glsdescwidth}}{%

5409 {\end{supertabular}}{%

Do nothing at the start of the table:

5410 \renewcommand\*{\glossaryheader}{}%

No group headings:

5411 \renewcommand\*{\glsgroupheading}[1]{}%

Main (level 0) entries put in a row (name in first column, description and page list in second column):

5412 \renewcommand\*{\glossaryentryfield}[5]{%

5413 \glstryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%

5414 \tabularnewline}{%

Sub entries put in a row (no name, description and page list in second column):

```
5415 \renewcommand*{\glossarysubentryfield}[6]{%
5416     &
5417     \glssubentryitem{##2}%
5418     \glstarget{##2}{\strut}##4\glspostdescription\space ##6%
5419     \tabularnewline}%
```

Blank row between groups:

```
5420 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & \tabularnewline\fi}%
5421 }
```

**superraggedborder** The superraggedborder style is like the above, but with horizontal and vertical lines:

```
5422 \newglossarystyle{superraggedborder}{%
```

Base it on the glostylesuperragged style:

```
5423 \glossarystyle{superragged}%
```

Put the glossary in a supertabular environment with two columns and a horizontal line in the head and tail:

```
5424 \renewenvironment{theglossary}%
5425     {\tablehead{\hline}\tabletail{\hline}%
5426     \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}|}%
5427     {\end{supertabular}}%
5428 }
```

**superraggedheader** The superraggedheader style is like the super style, but with a header:

```
5429 \newglossarystyle{superraggedheader}{%
```

Base it on the glostylesuperragged style:

```
5430 \glossarystyle{superragged}%
```

Put the glossary in a supertabular environment with two columns, a header and no tail:

```
5431 \renewenvironment{theglossary}%
5432     {\tablehead{\bfseries \entryname & \bfseries \descriptionname
5433     \tabularnewline}%
5434     \tabletail{}}%
5435     \begin{supertabular}{l>{\raggedright}p{\glsgdescwidth}}}%
5436     {\end{supertabular}}%
5437 }
```

**superraggedheaderborder** The superraggedheaderborder style is like the superragged style but with a header and border:

```
5438 \newglossarystyle{superraggedheaderborder}{%
```

Base it on the glostylesuper style:

```
5439 \glossarystyle{superragged}%
```

Put the glossary in a supertabular environment with two columns, a header and horizontal lines above and below the table:

```

5440 \renewenvironment{theglossary}%
5441   {\tablehead{\hline\bfseries \entryname &
5442     \bfseries \descriptionname\tabularnewline\hline}%
5443   \tabletail{\hline}
5444   \begin{supertabular}{|l|>{\raggedright}p{\glstdescwidth}|}%
5445   {\end{supertabular}}%
5446 }

```

**superragged3col** The superragged3col style is like the superragged style, but with 3 columns:

```

5447 \newglossarystyle{superragged3col}{%

```

Put the glossary in a supertabular environment with three columns and no head or tail:

```

5448 \renewenvironment{theglossary}%
5449   {\tablehead{}\tabletail{}}%
5450   \begin{supertabular}{|l>{\raggedright}p{\glstdescwidth}%
5451     >{\raggedright}p{\glspagelistwidth}}}%
5452   {\end{supertabular}}%

```

Do nothing at the start of the table:

```

5453 \renewcommand*{\glossaryheader}{}%

```

No group headings:

```

5454 \renewcommand*{\glsgroupheading}[1]{}%

```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```

5455 \renewcommand*{\glossaryentryfield}[5]{%
5456   \glstarget{##1}{\glstarget{##1}{##2} & ##3 & ##5\tabularnewline}%

```

Sub entries on a row (no name, description in second column, page list in last column):

```

5457 \renewcommand*{\glossarysubentryfield}[6]{%
5458   &
5459   \glssubentryitem{##2}%
5460   \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%

```

Blank row between groups:

```

5461 \renewcommand*{\glsgroupskip}{\ifglsgnোগroupskip\else & \tabularnewline\fi}%
5462 }

```

**superragged3colborder** The superragged3colborder style is like the superragged3col style, but with a border:

```

5463 \newglossarystyle{superragged3colborder}{%

```

Base it on the glostylesuperragged3col style:

```

5464 \glossarystyle{superragged3col}%

```

Put the glossary in a supertabular environment with three columns and a horizontal line in the head and tail:

```

5465 \renewenvironment{theglossary}%
5466   {\tablehead{\hline}\tabletail{\hline}%
5467    \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}|%
5468     >{\raggedright}p{\glspagelistwidth}|}%
5469   {\end{supertabular}}%
5470 }
```

**superragged3colheader** The superragged3colheader style is like the superragged3col style but with a header row:

```

5471 \newglossarystyle{superragged3colheader}{%
```

Base it on the glostylesuperragged3col style:

```

5472 \glossarystyle{superragged3col}%
```

Put the glossary in a supertabular environment with three columns, a header and no tail:

```

5473 \renewenvironment{theglossary}%
5474   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
5475    \bfseries\pagelistname\tabularnewline}\tabletail{}}%
5476   \begin{supertabular}{|l>{\raggedright}p{\glsgdescwidth}%
5477    >{\raggedright}p{\glspagelistwidth}}}%
5478   {\end{supertabular}}%
5479 }
```

**superragged3colheaderborder** The superragged3colheaderborder style is like the superragged3col style but with a header and border:

```

5480 \newglossarystyle{superragged3colheaderborder}{%
```

Base it on the glostylesuperragged3colborder style:

```

5481 \glossarystyle{superragged3colborder}%
```

Put the glossary in a supertabular environment with three columns, a header with horizontal lines and a horizontal line in the tail:

```

5482 \renewenvironment{theglossary}%
5483   {\tablehead{\hline
5484    \bfseries\entryname&\bfseries\descriptionname&
5485    \bfseries\pagelistname\tabularnewline\hline}%
5486   \tabletail{\hline}%
5487   \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}|%
5488    >{\raggedright}p{\glspagelistwidth}|}%
5489   {\end{supertabular}}%
5490 }
```

**altsuperragged4col** The altsuperragged4col glossary style is like altsuper4col style in the package but uses ragged right formatting in the description and page list columns.

```

5491 \newglossarystyle{altsuperragged4col}{%
```

Put the glossary in a supertabular environment with four columns and no head or tail:

```
5492 \renewenvironment{theglossary}%
5493   {\tablehead{}\tabletail{}}%
5494   \begin{supertabular}{1>{\raggedright}p{\glsgdescwidth}1%
5495     >{\raggedright}p{\glspagelistwidth}}}%
5496   {\end{supertabular}}%
```

Do nothing at the start of the table:

```
5497 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
5498 \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:

```
5499 \renewcommand*{\glossaryentryfield}[5]{%
5500   \glstarget{##1}{\glstarget{##1}{##2} & ##3 & ##4 & ##5\tabularnewline}%
```

Sub entries on a row with no name, the description in the second column, symbol in third column and page list in last column:

```
5501 \renewcommand*{\glossarysubentryfield}[6]{%
5502   &
5503   \glssubentryitem{##2}%
5504   \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%
```

Blank row between groups:

```
5505 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & & \tabularnewline\fi}%
5506 }
```

**perragged4colheader** The `altsuperragged4colheader` style is like the `altsuperragged4col` style but with a header row.

```
5507 \newglossarystyle{altsuperragged4colheader}{%
```

Base it on the `glostylealtsuperragged4col` style:

```
5508 \glossarystyle{altsuperragged4col}%
```

Put the glossary in a supertabular environment with four columns, a header and no tail:

```
5509 \renewenvironment{theglossary}%
5510   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
5511     \bfseries\symbolname &
5512     \bfseries\pagelistname\tabularnewline}\tabletail{}}%
5513   \begin{supertabular}{1>{\raggedright}p{\glsgdescwidth}1%
5514     >{\raggedright}p{\glspagelistwidth}}}%
5515   {\end{supertabular}}%
5516 }
```

**perragged4colborder** The `altsuperragged4colborder` style is like the `altsuperragged4col` style but with a border.

```
5517 \newglossarystyle{altsuperragged4colborder}{%
```

Base it on the `glostylealtsuperragged4col` style:

```
5518 \glossarystyle{altsuper4col}%
```

Put the glossary in a `supertabular` environment with four columns and a horizontal line in the head and tail:

```
5519 \renewenvironment{theglossary}%
5520 {\tablehead{\hline}\tabletail{\hline}%
5521 \begin{supertabular}%
5522   {\ll>{\raggedright}p{\glsgdescwidth}\ll%
5523    >{\raggedright}p{\glspagelistwidth}\l}%
5524 {\end{supertabular}}%
5525 }
```

`ged4colheaderborder` The `altsuperragged4colheaderborder` style is like the `altsuperragged4col` style but with a header and border.

```
5526 \newglossarystyle{altsuperragged4colheaderborder}{%
```

Base it on the `glostylealtsuperragged4col` style:

```
5527 \glossarystyle{altsuperragged4col}%
```

Put the glossary in a `supertabular` environment with four columns and a header bordered by horizontal lines and a horizontal line in the tail:

```
5528 \renewenvironment{theglossary}%
5529 {\tablehead{\hline
5530   \bfseries\entryname &
5531   \bfseries\descriptionname &
5532   \bfseries\symbolname &
5533   \bfseries\pagelistname\tabularnewline\hline}%
5534 \tabletail{\hline}%
5535 \begin{supertabular}%
5536   {\ll>{\raggedright}p{\glsgdescwidth}\ll%
5537    >{\raggedright}p{\glspagelistwidth}\l}%
5538 {\end{supertabular}}%
5539 }
```

### 3.9 Tree Styles (`glossary-tree.sty`)

The style file defines glossary styles that have a tree-like structure. These are designed for hierarchical glossaries.

```
5540 \ProvidesPackage{glossary-tree}[2012/09/21 v3.03 (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.

```
5541 \newglossarystyle{index}{%
```

Set the paragraph indentation and skip and define `\item` to be the same as that used by `theindex`:

```

5542 \renewenvironment{theglossary}%
5543   {\setlength{\parindent}{0pt}%
5544    \setlength{\parskip}{0pt plus 0.3pt}%
5545    \let\item\@idxitem}%
5546   {}%

```

Do nothing at the start of the environment:

```

5547 \renewcommand*{\glossaryheader}{}%

```

No group headers:

```

5548 \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.

```

5549 \renewcommand*{\glossaryentryfield}[5]{%
5550 \item\glsentryitem{##1}\textbf{\glstarget{##1}{##2}}%
5551 \ifx\relax##4\relax
5552 \else
5553   \space{##4}%
5554 \fi
5555 \space ##3\glspostdescription \space ##5}%

```

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 `\glossaryentryfield`, 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.

```

5556 \renewcommand*{\glossarysubentryfield}[6]{%
5557   \ifcase##1\relax
5558     % level 0
5559     \item
5560   \or
5561     % level 1
5562     \subitem
5563     \glssubentryitem{##2}%
5564   \else
5565     % all other levels
5566     \subsubitem
5567   \fi
5568   \textbf{\glstarget{##2}{##3}}%
5569   \ifx\relax##5\relax
5570   \else
5571     \space{##5}%
5572   \fi
5573   \space##4\glspostdescription\space ##6}%

```

Vertical gap between groups is the same as that used by indices:

```

5574 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}}

```

**indexgroup** The `indexgroup` style is like the `index` style but has headings.

```

5575 \newglossarystyle{indexgroup}{%

```

Base it on the `glostyleindex` style:

```
5576 \glossarystyle{index}%
```

Add a heading for each group. This puts the group's title in bold followed by a vertical gap.

```
5577 \renewcommand*{\glsgroupheading}[1]{%
5578   \item\textbf{\glsgetgrouptitle{##1}}\indexspace}%
5579 }
```

`indexhypergroup` The `indexhypergroup` style is like the `indexgroup` style but has hyper navigation.

```
5580 \newglossarystyle{indexhypergroup}{%
```

Base it on the `glostyleindex` style:

```
5581 \glossarystyle{index}%
```

Put navigation links to the groups at the start of the glossary:

```
5582 \renewcommand*{\glossaryheader}{%
5583   \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.

```
5584 \renewcommand*{\glsgroupheading}[1]{%
5585   \item\textbf{\glsnavhypertarget{##1}}{\glsgetgrouptitle{##1}}}%
5586   \indexspace}%
5587 }
```

`tree` The `tree` glossary style is similar in style to the `index` style, but can have arbitrary levels.

```
5588 \newglossarystyle{tree}{%
```

Set the paragraph indentation and skip:

```
5589 \renewenvironment{theglossary}%
5590   {\setlength{\parindent}{0pt}%
5591    \setlength{\parskip}{0pt plus 0.3pt}}%
5592   {}%
```

Do nothing at the start of the `theglossary` environment:

```
5593 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
5594 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries: name in bold, followed by symbol in brackets (if it exists), the description and the page list:

```
5595 \renewcommand{\glossaryentryfield}[5]{%
5596   \hangindent0pt\relax
5597   \parindent0pt\relax
5598   \glsentryitem{##1}\textbf{\glstarget{##1}{##2}}%
5599   \ifx\relax##4\relax
5600   \else
5601     \space{##4}%
5602   \fi
5603   \space ##3\glspostdescription \space ##5\par}%
```



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.

```
5604 \renewcommand{\glossarysubentryfield}[6]{%
5605   \hangindent##1\glstreeindent\relax
5606   \parindent##1\glstreeindent\relax
5607   \ifnum##1=1\relax
5608     \glssubentryitem{##2}%
5609   \fi
5610   \textbf{\glstarget{##2}{##3}}%
5611   \ifx\relax##5\relax
5612   \else
5613     \space{##5}%
5614   \fi
5615   \space##4\glspostdescription\space ##6\par}%
```

Vertical gap between groups is the same as that used by indices:

```
5616 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}}
```

**treegroup** Like the tree style but the glossary groups have headings.

```
5617 \newglossarystyle{treegroup}{%
```

Base it on the `glostyletree` style:

```
5618 \glossarystyle{tree}%
```

Each group has a heading (in bold) followed by a vertical gap):

```
5619 \renewcommand{\glsgroupheading}[1]{\par
5620   \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
5621 }
```

**treehypergroup** The `treehypergroup` style is like the `treegroup` style, but has a set of links to the groups at the start of the glossary.

```
5622 \newglossarystyle{treehypergroup}{%
```

Base it on the `glostyletree` style:

```
5623 \glossarystyle{tree}%
```

Put navigation links to the groups at the start of the `theglossary` environment:

```
5624 \renewcommand*{\glossaryheader}{%
5625   \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
5626 \renewcommand*{\glsgroupheading}[1]{%
5627   \par\noindent
5628   \textbf{\glsnavigationtarget{##1}{\glsgrouptitle{##1}}}\par
5629   \indexspace}%
5630 }
```

**\glstreeindent** Length governing left indent for each level of the tree style.

```
5631 \newlength\glstreeindent
5632 \setlength{\glstreeindent}{10pt}
```

**treenoname** The **treenoname** glossary style is like the **tree** style, but doesn't print the name or symbol for sub-levels.

```
5633 \newglossarystyle{treenoname}{%
```

Set the paragraph indentation and skip:

```
5634 \renewenvironment{theglossary}%
5635 {\setlength{\parindent}{0pt}%
5636 \setlength{\parskip}{0pt plus 0.3pt}}%
5637 {}%
```

No header:

```
5638 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
5639 \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.

```
5640 \renewcommand{\glossaryentryfield}[5]{%
5641 \hangindent0pt\relax
5642 \parindent0pt\relax
5643 \glstarget{##1}\textbf{\glstarget{##1}{##2}}%
5644 \ifx\relax##4\relax
5645 \else
5646 \space{##4}%
5647 \fi
5648 \space ##3\glspostdescription \space ##5\par}%
```

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.

```
5649 \renewcommand{\glossarysubentryfield}[6]{%
5650 \hangindent##1\glstreeindent\relax
5651 \parindent##1\glstreeindent\relax
5652 \ifnum##1=1\relax
5653 \glssubentryitem{##2}%
5654 \fi
5655 \glstarget{##2}{\strut}%
5656 ##4\glspostdescription\space ##6\par}%
```

Vertical gap between groups is the same as that used by indices:

```
5657 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
5658 }
```

**treenonamegroup** Like the **treenoname** style but the glossary groups have headings.

```
5659 \newglossarystyle{treenonamegroup}{%
```

Base it on the `glostyletreenoname` style:

```
5660 \glossarystyle{treenoname}%
```

Give each group a heading:

```
5661 \renewcommand{\glsgroupheading}[1]{\par
5662 \noindent\textbf{\glsgetgrouptitle{##1}}\par\indexspace}%
5663 }
```

`treenonamehypergroup` The `treenonamehypergroup` style is like the `treenonamegroup` style, but has a set of links to the groups at the start of the glossary.

```
5664 \newglossarystyle{treenonamehypergroup}{%
```

Base it on the `glostytreetreenoname` style:

```
5665 \glossarystyle{treenoname}%
```

Put navigation links to the groups at the start of the `theglossary` environment:

```
5666 \renewcommand*{\glossaryheader}{%
```

```
5667 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
5668 \renewcommand*{\glsgroupheading}[1]{%
```

```
5669 \par\noindent
```

```
5670 \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
```

```
5671 \indexspace}%
```

```
5672 }
```

`\glissetwidest` `\glissetwidest[<level>]{<text>}` sets the widest text for the given level. It is used by the `alttree` glossary styles to determine the indentation of each level.

```
5673 \newcommand*{\glissetwidest}[2][0]{%
```

```
5674 \expandafter\def\csname @glswidestname\romannumeral#1\endcsname{%
```

```
5675 #2}%
```

```
5676 }
```

`\@glswidestname` Initialise `\@glswidestname`.

```
5677 \newcommand*{\@glswidestname}{}
```

`alttree` The `alttree` glossary style is similar in style to the `tree` style, but the indentation is obtained from the width of `\@glswidestname` which is set using `\glissetwidest`.

```
5678 \newglossarystyle{alttree}{%
```

Redefine the `theglossary` environment.

```
5679 \renewenvironment{theglossary}%
```

```
5680 {\def\@gls@prevlevel{-1}%
```

```
5681 \mbox{}\par}%
```

```
5682 {\par}%
```

Set the header and group headers to nothing.

```
5683 \renewcommand*{\glossaryheader}{}%
```

```
5684 \renewcommand*{\glsgroupheading}[1]{}%
```

Redefine the way that the level 0 entries are displayed.

```
5685 \renewcommand{\glossaryentryfield}[5]{%
```

If the level hasn't changed, keep the same settings, otherwise change `\glstreeindent` accordingly.

```
5686 \ifnum\@gls@prevlevel=0\relax
```

```
5687 \else
```

Find out how big the indentation should be by measuring the widest entry.

```

5688      \settowidth{\glstreeindent}{\textbf{\@glswidestname\space}}%

```

Set the hangindent and paragraph indent.

```

5689      \hangindent\glstreeindent
5690      \parindent\glstreeindent
5691      \fi

```

Put the name to the left of the paragraph block.

```

5692      \makebox[0pt][r]{\makebox[\glstreeindent][l]{%
5693        \glstryitem{##1}\textbf{\glstarget{##1}{##2}}}%

```

If the symbol is missing, ignore it, otherwise put it in brackets.

```

5694      \ifx\relax##4\relax
5695      \else
5696        (##4)\space
5697      \fi

```

Do the description followed by the description terminator and location list.

```

5698      ##3\glspostdescription \space ##5\par

```

Set the previous level to 0.

```

5699      \def\@gls@prevlevel{0}%
5700      }%

```

Redefine the way sub-entries are displayed.

```

5701      \renewcommand{\glossarysubentryfield}[6]{%

```

Increment and display the sub-entry counter if this is a level 1 entry and the sub-entry counter is in use.

```

5702      \ifnum##1=1\relax
5703        \glssubentryitem{##2}%
5704      \fi

```

If the level hasn't changed, keep the same settings, otherwise adjust \glstreeindent accordingly.

```

5705      \ifnum\@gls@prevlevel=##1\relax
5706      \else

```

Compute the widest entry for this level, or for level 0 if not defined for this level.  
Store in \gls@tmplen

```

5707      \@ifundefined{\@glswidestname\romannumeral##1}{%
5708        \settowidth{\gls@tmplen}{\textbf{\@glswidestname\space}}{%
5709        \settowidth{\gls@tmplen}{\textbf{%
5710          \csname \@glswidestname\romannumeral##1\endcsname\space}}}%

```

Determine if going up or down a level

```

5711      \ifnum\@gls@prevlevel<##1\relax

```

Depth has increased, so add the width of the widest entry to \glstreeindent.

```

5712        \setlength\glstreeindent\gls@tmplen
5713        \addtolength\glstreeindent\parindent
5714        \parindent\glstreeindent
5715      \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`.

```

5716      \ifundefined{@glswidestname\romannumeral\@gls@prevlevel}{%
5717      \settowidth{\glstreeindent}{\textbf{%
5718      \@glswidestname\space}}}{%
5719      \settowidth{\glstreeindent}{\textbf{%
5720      \csname @glswidestname\romannumeral\@gls@prevlevel
5721      \endcsname\space}}}{%

```

Subtract this length from the previous level's paragraph indent and set to `\glstreeindent`.

```

5722      \addtolength\parindent{-\glstreeindent}%
5723      \setlength\glstreeindent\parindent
5724      \fi
5725      \fi

```

Set the hanging indentation.

```

5726      \hangindent\glstreeindent

```

Put the name to the left of the paragraph block

```

5727      \makebox[0pt][r]{\makebox[\@gls@tmplen][l]{%
5728      \textbf{\glstarget{##2}{##3}}}}}%

```

If the symbol is missing, ignore it, otherwise put it in brackets.

```

5729      \ifx##5\relax\relax
5730      \else
5731      (##5)\space
5732      \fi

```

Do the description followed by the description terminator and location list.

```

5733      ##4\glspostdescription\space ##6\par

```

Set the previous level macro to the current level.

```

5734      \def\@gls@prevlevel{##1}%
5735      }%

```

Vertical gap between groups is the same as that used by indices:

```

5736      \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
5737 }

```

**almtreegroup** Like the almtree style but the glossary groups have headings.

```

5738 \newglossarystyle{almtreegroup}{%

```

Base it on the `glostylealmtree` style:

```

5739 \glossarystyle{almtree}%

```

Give each group a heading.

```

5740 \renewcommand{\glsgroupheading}[1]{\par
5741 \def\@gls@prevlevel{-1}%
5742 \hangindent0pt\relax
5743 \parindent0pt\relax

```

```

5744 \textbf{\glsgetgrouptitle{##1}}\par\indexspace}%
5745 }

```

`almtreehypergroup` The `almtreehypergroup` style is like the `almtreegroup` style, but has a set of links to the groups at the start of the glossary.

```

5746 \newglossarystyle{almtreehypergroup}{%

```

Base it on the `glostylealmtree` style:

```

5747 \glossarystyle{almtree}%

```

Put the navigation links in the header

```

5748 \renewcommand*{\glossaryheader}{%
5749 \par
5750 \def\@gls@prevlevel{-1}%
5751 \hangindent0pt\relax
5752 \parindent0pt\relax
5753 \textbf{\glsnavigation}\par\indexspace}%

```

Put a hypertarget at the start of each group

```

5754 \renewcommand*{\glsgroupheading}[1]{%
5755 \par
5756 \def\@gls@prevlevel{-1}%
5757 \hangindent0pt\relax
5758 \parindent0pt\relax
5759 \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
5760 \indexspace}}

```

## 4 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.

```

5761 \NeedsTeXFormat{LaTeX2e}
5762 \ProvidesPackage{glossaries-compatible-207}[2011/04/02 v1.0 (NLCT)]

```

`\GlsAddXdyAttribute` Adds an attribute in old format.

```

5763 \ifglsxindy
5764 \renewcommand*\GlsAddXdyAttribute[1]{%
5765 \edef\@xdyattributes{\@xdyattributes ^^J \string"#1\string"}%
5766 \expandafter\toks@\expandafter{\@xdylocref}%
5767 \edef\@xdylocref{\the\toks@ ^^J%
5768 (markup-locref
5769 :open \string"\string~n\string\setentrycounter
5770 {\noexpand\glscounter}%
5771 \expandafter\string\csname#1\endcsname
5772 \expandafter@gobble\string\{\string" ^^J
5773 :close \string"\expandafter@gobble\string\}\string" ^^J
5774 :attr \string"#1\string")}}

```

Only has an effect before \writeist:

5775 \fi

\GlsAddXdyCounters

```
5776 \renewcommand*\GlsAddXdyCounters[1]{%
5777   \GlossariesWarning{\string\GlsAddXdyCounters\space not available
5778     in compatibility mode.}%
5779 }
```

Add predefined attributes

```
5780 \GlsAddXdyAttribute{glsnumberformat}
5781 \GlsAddXdyAttribute{textrm}
5782 \GlsAddXdyAttribute{textsf}
5783 \GlsAddXdyAttribute{texttt}
5784 \GlsAddXdyAttribute{textbf}
5785 \GlsAddXdyAttribute{textmd}
5786 \GlsAddXdyAttribute{textit}
5787 \GlsAddXdyAttribute{textup}
5788 \GlsAddXdyAttribute{textsl}
5789 \GlsAddXdyAttribute{textsc}
5790 \GlsAddXdyAttribute{emph}
5791 \GlsAddXdyAttribute{glshypernumber}
5792 \GlsAddXdyAttribute{hyperarm}
5793 \GlsAddXdyAttribute{hypersf}
5794 \GlsAddXdyAttribute{hypertt}
5795 \GlsAddXdyAttribute{hyperbf}
5796 \GlsAddXdyAttribute{hypermd}
5797 \GlsAddXdyAttribute{hyperit}
5798 \GlsAddXdyAttribute{hyperup}
5799 \GlsAddXdyAttribute{hypersl}
5800 \GlsAddXdyAttribute{hypersc}
5801 \GlsAddXdyAttribute{hyperemph}
```

\GlsAddXdyLocation Restore v2.07 definition:

```
5802 \ifglxindy
5803   \renewcommand*\GlsAddXdyLocation[2]{%
5804     \edef\@xdyuserlocationdefs{%
5805       \@xdyuserlocationdefs ^^J%
5806       (define-location-class \string"#1\string"^^J\space\space
5807         \space(#2))
5808     }%
5809     \edef\@xdyuserlocationnames{%
5810       \@xdyuserlocationnames^^J\space\space\space
5811       \string"#1\string"%
5812     }
5813 \fi
```

\@do@wrglossary

```
5814 \renewcommand*\@do@wrglossary[1]{%
```

Determine whether to use xindy or makeindex syntax

```
5815 \ifglxsindy
```

Need to determine if the formatting information starts with a ( or ) indicating a range.

```
5816 \expandafter\@glo@check@mkidxrangechar\@glsnumberformat\@nil
5817 \def\@glo@range{}%
5818 \expandafter\if\@glo@prefix(\relax
5819 \def\@glo@range{:open-range}%
5820 \else
5821 \expandafter\if\@glo@prefix)\relax
5822 \def\@glo@range{:close-range}%
5823 \fi
5824 \fi
```

Get the location and escape any special characters

```
5825 \protected@edef\@glslocref{\theglentrycounter}%
5826 \@gls@checkmkidxchars\@glslocref
```

Write to the glossary file using xindy syntax.

```
5827 \glossary[\csname glo@#1@type\endcsname]{%
5828 (indexentry :tkey (\csname glo@#1@index\endcsname)
5829 :locref \string"\@glslocref\string" %
5830 :attr \string"\@glo@suffix\string" \@glo@range
5831 )
5832 }%
5833 \else
```

Convert the format information into the format required for makeindex

```
5834 \@set@glo@numformat\@glo@numfmt\@gls@counter\@glsnumberformat
```

Write to the glossary file using makeindex syntax.

```
5835 \glossary[\csname glo@#1@type\endcsname]{%
5836 \string\glossaryentry{\csname glo@#1@index\endcsname
5837 \@gls@encapchar\@glo@numfmt}{\theglentrycounter}}%
5838 \fi
5839 }
```

\@set@glo@numformat Only had 3 arguments in v2.07

```
5840 \def\@set@glo@numformat#1#2#3{%
5841 \expandafter\@glo@check@mkidxrangechar#3\@nil
5842 \protected@edef#1{%
5843 \@glo@prefix setentrycounter[] {#2}%
5844 \expandafter\string\csname\@glo@suffix\endcsname
5845 }%
5846 \@gls@checkmkidxchars#1%
5847 }
```

\writeist Redefine \writeist back to the way it was in v2.07, but change \istfile to \glswrite.



```

5848 \ifglxindy
5849   \def\writeist{%
5850     \openout\glswrite=\istfilename
5851     \write\glswrite{;; xindy style file created by the glossaries
5852       package in compatible-2.07 mode}%
5853     \write\glswrite{;; for document '\jobname' on
5854       \the\year-\the\month-\the\day}%
5855     \write\glswrite{^^J; required styles^^J}
5856     \@for\@xdystyle:=\@xdyrequiredstyles\do{%
5857       \ifx\@xdystyle\@empty
5858       \else
5859         \protected@write\glswrite{{(require
5860           \string"\@xdystyle.xdy\string")}}%
5861       \fi
5862     }%
5863     \write\glswrite{^^J%
5864       ; list of allowed attributes (number formats)^^J}%
5865     \write\glswrite{(define-attributes ((\@xdyattributes)))}%
5866     \write\glswrite{^^J; user defined alphabets^^J}%
5867     \write\glswrite{\@xdyuseralphabets}%
5868     \write\glswrite{^^J; location class definitions^^J}%
5869     \protected@edef\@gls@roman{\@roman{0}\string"
5870       \string"roman-numbers-lowercase\string" :sep \string"}%
5871     \@onelevel@sanitize\@gls@roman
5872     \edef\@tmp{\string" \string"roman-numbers-lowercase\string"
5873       :sep \string"}%
5874     \@onelevel@sanitize\@tmp
5875     \ifx\@tmp\@gls@roman
5876       \write\glswrite{(define-location-class
5877         \string"roman-page-numbers\string"^^J\space\space\space
5878         (\string"roman-numbers-lowercase\string")
5879         :min-range-length \@glsminrange)}}%
5880     \else
5881       \write\glswrite{(define-location-class
5882         \string"roman-page-numbers\string"^^J\space\space\space
5883         (:sep "\@gls@roman")
5884         :min-range-length \@glsminrange)}}%
5885     \fi
5886     \write\glswrite{(define-location-class
5887       \string"Roman-page-numbers\string"^^J\space\space\space
5888       (\string"roman-numbers-uppercase\string")
5889       :min-range-length \@glsminrange)}}%
5890     \write\glswrite{(define-location-class
5891       \string"arabic-page-numbers\string"^^J\space\space\space
5892       (\string"arabic-numbers\string")
5893       :min-range-length \@glsminrange)}}%
5894     \write\glswrite{(define-location-class
5895       \string"alpha-page-numbers\string"^^J\space\space\space
5896       (\string"alpha\string")

```

```

5897         :min-range-length \@glsminrange))}%
5898 \write\glswrite{(define-location-class
5899   \string"Alpha-page-numbers\string"^^J\space\space\space
5900   (\string"ALPHA\string")
5901   :min-range-length \@glsminrange))}%
5902 \write\glswrite{(define-location-class
5903   \string"Appendix-page-numbers\string"^^J\space\space\space
5904   (\string"ALPHA\string"
5905   :sep \string"\@glsAlphacompositor\string"
5906   \string"arabic-numbers\string")
5907   :min-range-length \@glsminrange))}%
5908 \write\glswrite{(define-location-class
5909   \string"arabic-section-numbers\string"^^J\space\space\space
5910   (\string"arabic-numbers\string"
5911   :sep \string"\glscompositor\string"
5912   \string"arabic-numbers\string")
5913   :min-range-length \@glsminrange))}%
5914 \write\glswrite{^^J; user defined location classes}%
5915 \write\glswrite{@xdyuserlocationdefs}%
5916 \write\glswrite{^^J; define cross-reference class^^J}%
5917 \write\glswrite{(define-crossref-class \string"see\string"
5918   :unverified )}%
5919 \write\glswrite{(markup-crossref-list
5920   :class \string"see\string"^^J\space\space\space
5921   :open \string"\string\glsseeformat\string"
5922   :close \string"{}\string")}%
5923 \write\glswrite{^^J; define the order of the location classes}%
5924 \write\glswrite{(define-location-class-order
5925   (\@xdylocationclassorder))}%
5926 \write\glswrite{^^J; define the glossary markup^^J}%
5927 \write\glswrite{(markup-index^^J\space\space\space
5928   :open \string"\string
5929   \glossarysection[\string\glossarytoctitle]{\string
5930   \glossarytitle}\string\glossarypreamble\string~n\string\begin
5931   {theglossary}\string\glossaryheader\string~n\string" ^^J\space
5932   \space\space:close \string"\expandafter\@gobble
5933   \string%\string~n\string
5934   \end{theglossary}\string\glossarypostamble
5935   \string~n\string" ^^J\space\space\space
5936   :tree)}}%
5937 \write\glswrite{(markup-letter-group-list
5938   :sep \string"\string\glsgroupskip\string~n\string")}%
5939 \write\glswrite{(markup-indexentry
5940   :open \string"\string\relax \string\glsresetentrylist
5941   \string~n\string")}%
5942 \write\glswrite{(markup-locclass-list :open
5943   \string"\glsopenbrace\string\glossaryentrynumbers
5944   \glsopenbrace\string\relax\space \string"^^J\space\space\space
5945   :sep \string", \string"

```

```

5946 :close \string"\glsclosebrace\glsclosebrace\string"))}%
5947 \write\glswrite{(markup-locoref-list
5948 :sep \string"\string\delimN\space\string"))}%
5949 \write\glswrite{(markup-range
5950 :sep \string"\string\delimR\space\string"))}%
5951 \@onelevel@sanitize\gls@suffixF
5952 \@onelevel@sanitize\gls@suffixFF
5953 \ifx\gls@suffixF\@empty
5954 \else
5955 \write\glswrite{(markup-range
5956 :close "\gls@suffixF" :length 1 :ignore-end))}%
5957 \fi
5958 \ifx\gls@suffixFF\@empty
5959 \else
5960 \write\glswrite{(markup-range
5961 :close "\gls@suffixFF" :length 2 :ignore-end))}%
5962 \fi
5963 \write\glswrite{^^J; define format to use for locations^^J}%
5964 \write\glswrite{@xdylocoref}%
5965 \write\glswrite{^^J; define letter group list format^^J}%
5966 \write\glswrite{(markup-letter-group-list
5967 :sep \string"\string\glsgroupskip\string~n\string"))}%
5968 \write\glswrite{^^J; letter group headings^^J}%
5969 \write\glswrite{(markup-letter-group
5970 :open-head \string"\string\glsgroupheading
5971 \glsopenbrace\string"^^J\space\space\space
5972 :close-head \string"\glsclosebrace\string"))}%
5973 \write\glswrite{^^J; additional letter groups^^J}%
5974 \write\glswrite{@xdylettergroups}%
5975 \write\glswrite{^^J; additional sort rules^^J}%
5976 \write\glswrite{@xdysortrules}%
5977 \noist}
5978 \else
5979 \edef\@gls@actualchar{\string?}
5980 \edef\@gls@encapchar{\string|}
5981 \edef\@gls@levelchar{\string!}
5982 \edef\@gls@quotechar{\string"}
5983 \def\writeist{\relax
5984 \openout\glswrite=\istfilename
5985 \write\glswrite{\expandafter\@gobble\string\% makeindex style file
5986 created by the glossaries package}
5987 \write\glswrite{\expandafter\@gobble\string\% for document
5988 '\jobname' on \the\year-\the\month-\the\day}
5989 \write\glswrite{actual '\@gls@actualchar'}
5990 \write\glswrite{encap '\@gls@encapchar'}
5991 \write\glswrite{level '\@gls@levelchar'}
5992 \write\glswrite{quote '\@gls@quotechar'}
5993 \write\glswrite{keyword \string"\string\glossaryentry\string"}
5994 \write\glswrite{preamble \string"\string\glossarysection[\string

```

```

5995     \glossarytoctitle]{\string\glossarytitle}\string
5996     \glossarypreamble\string\n\string\begin{theglossary}\string
5997     \glossaryheader\string\n\string"}
5998 \write\glswrite{postamble \string"\string%\string\n\string
5999     \end{theglossary}\string\glossarypostamble\string\n
6000     \string"}
6001 \write\glswrite{group_skip \string"\string\glsgroupskip\string\n
6002     \string"}
6003 \write\glswrite{item_0 \string"\string%\string\n\string"}
6004 \write\glswrite{item_1 \string"\string%\string\n\string"}
6005 \write\glswrite{item_2 \string"\string%\string\n\string"}
6006 \write\glswrite{item_01 \string"\string%\string\n\string"}
6007 \write\glswrite{item_x1
6008     \string"\string\relax \string\glsglresetentrylist\string\n
6009     \string"}
6010 \write\glswrite{item_12 \string"\string%\string\n\string"}
6011 \write\glswrite{item_x2
6012     \string"\string\relax \string\glsglresetentrylist\string\n
6013     \string"}
6014 \write\glswrite{delim_0 \string"\string\{\string
6015     \glossaryentrynumbers\string\{\string\relax \string"}
6016 \write\glswrite{delim_1 \string"\string\{\string
6017     \glossaryentrynumbers\string\{\string\relax \string"}
6018 \write\glswrite{delim_2 \string"\string\{\string
6019     \glossaryentrynumbers\string\{\string\relax \string"}
6020 \write\glswrite{delim_t \string"\string\}\string\}\string"}
6021 \write\glswrite{delim_n \string"\string\delimN \string"}
6022 \write\glswrite{delim_r \string"\string\delimR \string"}
6023 \write\glswrite{headings_flag 1}
6024 \write\glswrite{heading_prefix
6025     \string"\string\glsgroupheading\string\{\string"}
6026 \write\glswrite{heading_suffix
6027     \string"\string\}\string\relax
6028     \string\glsglresetentrylist \string"}
6029 \write\glswrite{symhead_positive \string"glssymbols\string"}
6030 \write\glswrite{numhead_positive \string"glsglnumbers\string"}
6031 \write\glswrite{page_compositor \string"glsglcompositor\string"}
6032 \@glsglscbsdq\glsgl@suffixF
6033 \@glsglscbsdq\glsgl@suffixFF
6034 \ifx\glsgl@suffixF\@empty
6035 \else
6036     \write\glswrite{suffix_2p \string"\glsgl@suffixF\string"}
6037 \fi
6038 \ifx\glsgl@suffixFF\@empty
6039 \else
6040     \write\glswrite{suffix_3p \string"\glsgl@suffixFF\string"}
6041 \fi
6042 \noist
6043 }

```

```
6044 \fi
```

```
\noist
```

```
6045 \renewcommand*{\noist}{\let\writeist\relax}
```

## 5 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.

```
6046 \NeedsTeXFormat{LaTeX2e}
```

Package version number now in line with main glossaries package number but will only be updated when `glossaries-accsupp.sty` is modified.

```
6047 \ProvidesPackage{glossaries-accsupp}[2011/04/02 v3.0 (NLCT)]
```

```
6048 Experimental glossaries accessibility
```

Pass all options to glossaries:

```
6049 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{glossaries}}
```

Process options:

```
6050 \ProcessOptions
```

Required packages:

```
6051 \RequirePackage{glossaries}
```

```
6052 \RequirePackage{accsupp}
```

### 5.1 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:

```
6053 \define@key{glossentry}{access}{%
```

```
6054 \def\@glo@access{#1}%
```

```
6055 }
```

**textaccess** The replacement text corresponding to the text key:

```
6056 \define@key{glossentry}{textaccess}{%
```

```
6057 \def\@glo@textaccess{#1}%
```

```
6058 }
```

**firstaccess** The replacement text corresponding to the first key:

```
6059 \define@key{glossentry}{firstaccess}{%
```

```
6060 \def\@glo@firstaccess{#1}%
```

```
6061 }
```

pluralaccess    The replacement text corresponding to the plural key:

```
6062 \define@key{glossentry}{pluralaccess}{%
6063   \def\@glo@pluralaccess{#1}%
6064 }
```

firstpluralaccess    The replacement text corresponding to the firstplural key:

```
6065 \define@key{glossentry}{firstpluralaccess}{%
6066   \def\@glo@firstpluralaccess{#1}%
6067 }
```

symbolaccess    The replacement text corresponding to the symbol key:

```
6068 \define@key{glossentry}{symbolaccess}{%
6069   \def\@glo@symbolaccess{#1}%
6070 }
```

symbolpluralaccess    The replacement text corresponding to the symbolplural key:

```
6071 \define@key{glossentry}{symbolpluralaccess}{%
6072   \def\@glo@symbolpluralaccess{#1}%
6073 }
```

descriptionaccess    The replacement text corresponding to the description key:

```
6074 \define@key{glossentry}{descriptionaccess}{%
6075   \def\@glo@descaccess{#1}%
6076 }
```

descriptionpluralaccess    The replacement text corresponding to the descriptionplural key:

```
6077 \define@key{glossentry}{descriptionpluralaccess}{%
6078   \def\@glo@descpluralaccess{#1}%
6079 }
```

shortaccess    The replacement text corresponding to the short key:

```
6080 \define@key{glossentry}{shortaccess}{%
6081   \def\@glo@shortaccess{#1}%
6082 }
```

shortpluralaccess    The replacement text corresponding to the shortplural key:

```
6083 \define@key{glossentry}{shortpluralaccess}{%
6084   \def\@glo@shortpluralaccess{#1}%
6085 }
```

longaccess    The replacement text corresponding to the long key:

```
6086 \define@key{glossentry}{longaccess}{%
6087   \def\@glo@longaccess{#1}%
6088 }
```

longpluralaccess    The replacement text corresponding to the longplural key:

```
6089 \define@key{glossentry}{longpluralaccess}{%
6090   \def\@glo@longpluralaccess{#1}%
6091 }
```

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={\glsacssupp{inches}{in}}.

\@gls@noaccess Indicates that no replacement text has been provided.

```
6092 \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):

```
6093 \let\@gls@oldnewglossaryentryprehook\@newglossaryentryprehook
6094 \renewcommand*{\@newglossaryentryprehook}{%
6095   \@gls@oldnewglossaryentryprehook
6096   \def\@glo@access{\@glo@symbol}%
```

Initialise the other keys:

```
6097   \def\@glo@textaccess{\@glo@access}%
6098   \def\@glo@firstaccess{\@glo@access}%
6099   \def\@glo@pluralaccess{\@glo@textaccess}%
6100   \def\@glo@firstpluralaccess{\@glo@pluralaccess}%
6101   \def\@glo@symbolaccess{\relax}%
6102   \def\@glo@symbolpluralaccess{\@glo@symbolaccess}%
6103   \def\@glo@descaccess{\relax}%
6104   \def\@glo@descpluralaccess{\@glo@descaccess}%
6105   \def\@glo@shortaccess{\relax}%
6106   \def\@glo@shortpluralaccess{\@glo@shortaccess}%
6107   \def\@glo@longaccess{\relax}%
6108   \def\@glo@longpluralaccess{\@glo@longaccess}%
6109 }
```

Add to the end hook:

```
6110 \let\@gls@oldnewglossaryentryposthook\@newglossaryentryposthook
6111 \renewcommand*{\@newglossaryentryposthook}{%
6112   \@gls@oldnewglossaryentryposthook
```

Store the access information:

```
6113   \expandafter
6114     \protected@xdef\csname glo@\@glo@label @access\endcsname{%
6115       \@glo@access}%
6116   \expandafter
6117     \protected@xdef\csname glo@\@glo@label @textaccess\endcsname{%
6118       \@glo@textaccess}%
6119   \expandafter
6120     \protected@xdef\csname glo@\@glo@label @firstaccess\endcsname{%
6121       \@glo@firstaccess}%
6122   \expandafter
6123     \protected@xdef\csname glo@\@glo@label @pluralaccess\endcsname{%
6124       \@glo@pluralaccess}%
6125   \expandafter
6126     \protected@xdef\csname glo@\@glo@label @firstpluralaccess\endcsname{%
6127       \@glo@firstpluralaccess}%
6128   \expandafter
```

```

6129 \protected@xdef\csname glo@\@glo@label @symbolaccess\endcsname{%
6130 \@glo@symbolaccess}%
6131 \expandafter
6132 \protected@xdef\csname glo@\@glo@label @symbolpluralaccess\endcsname{%
6133 \@glo@symbolpluralaccess}%
6134 \expandafter
6135 \protected@xdef\csname glo@\@glo@label @descaccess\endcsname{%
6136 \@glo@descaccess}%
6137 \expandafter
6138 \protected@xdef\csname glo@\@glo@label @descpluralaccess\endcsname{%
6139 \@glo@descpluralaccess}%
6140 \expandafter
6141 \protected@xdef\csname glo@\@glo@label @shortaccess\endcsname{%
6142 \@glo@shortaccess}%
6143 \expandafter
6144 \protected@xdef\csname glo@\@glo@label @shortpluralaccess\endcsname{%
6145 \@glo@shortpluralaccess}%
6146 \expandafter
6147 \protected@xdef\csname glo@\@glo@label @longaccess\endcsname{%
6148 \@glo@longaccess}%
6149 \expandafter
6150 \protected@xdef\csname glo@\@glo@label @longpluralaccess\endcsname{%
6151 \@glo@longpluralaccess}%
6152 }

```

## 5.2 Accessing Replacement Text

`\glsentryaccess` Get the value of the access key for the entry with the given label:

```

6153 \newcommand*{\glsentryaccess}[1]{%
6154 \csname glo@#1@access\endcsname
6155 }

```

`\glsentrytextaccess` Get the value of the textaccess key for the entry with the given label:

```

6156 \newcommand*{\glsentrytextaccess}[1]{%
6157 \csname glo@#1@textaccess\endcsname
6158 }

```

`\glsentryfirstaccess` Get the value of the firstaccess key for the entry with the given label:

```

6159 \newcommand*{\glsentryfirstaccess}[1]{%
6160 \csname glo@#1@firstaccess\endcsname
6161 }

```

`\glsentrypluralaccess` Get the value of the pluralaccess key for the entry with the given label:

```

6162 \newcommand*{\glsentrypluralaccess}[1]{%
6163 \csname glo@#1@pluralaccess\endcsname
6164 }

```

`\glsentryfirstpluralaccess` Get the value of the firstpluralaccess key for the entry with the given label:



```

6165 \newcommand*{\glsentryfirstpluralaccess}[1]{%
6166   \csname glo@#1@firstpluralaccess\endcsname
6167 }

```

`\glsentrysymbolaccess` Get the value of the `symbolaccess` key for the entry with the given label:

```

6168 \newcommand*{\glsentrysymbolaccess}[1]{%
6169   \csname glo@#1@symbolaccess\endcsname
6170 }

```

`\glsentrysymbolpluralaccess` Get the value of the `symbolpluralaccess` key for the entry with the given label:

```

6171 \newcommand*{\glsentrysymbolpluralaccess}[1]{%
6172   \csname glo@#1@symbolpluralaccess\endcsname
6173 }

```

`\glsentrydescaccess` Get the value of the `descriptionaccess` key for the entry with the given label:

```

6174 \newcommand*{\glsentrydescaccess}[1]{%
6175   \csname glo@#1@descaccess\endcsname
6176 }

```

`\glsentrydescpluralaccess` Get the value of the `descriptionpluralaccess` key for the entry with the given label:

```

6177 \newcommand*{\glsentrydescpluralaccess}[1]{%
6178   \csname glo@#1@descaccess\endcsname
6179 }

```

`\glsentryshortaccess` Get the value of the `shortaccess` key for the entry with the given label:

```

6180 \newcommand*{\glsentryshortaccess}[1]{%
6181   \csname glo@#1@shortaccess\endcsname
6182 }

```

`\glsentryshortpluralaccess` Get the value of the `shortpluralaccess` key for the entry with the given label:

```

6183 \newcommand*{\glsentryshortpluralaccess}[1]{%
6184   \csname glo@#1@shortpluralaccess\endcsname
6185 }

```

`\glsentrylongaccess` Get the value of the `longaccess` key for the entry with the given label:

```

6186 \newcommand*{\glsentrylongaccess}[1]{%
6187   \csname glo@#1@longaccess\endcsname
6188 }

```

`\glsentrylongpluralaccess` Get the value of the `longpluralaccess` key for the entry with the given label:

```

6189 \newcommand*{\glsentrylongpluralaccess}[1]{%
6190   \csname glo@#1@longpluralaccess\endcsname
6191 }

```

`\glsacccsupp` `\glsacccsupp{<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.)

```
6192 \newcommand*{\glsaccsupp}[2]{%
6193   \BeginAccSupp{ActualText=#1}#2\EndAccSupp{}%
6194 }
```

`\xglsaccsupp` Fully expands replacement text before calling `\glsaccsupp`

```
6195 \newcommand*{\xglsaccsupp}[2]{%
6196   \protected@edef\@gls@replacementtext{#1}%
6197   \expandafter\xglsaccsupp\expandafter{\@gls@replacementtext}{#2}%
6198 }
```

`\glsnameaccessdisplay` Displays the first argument with the accessibility text for the entry with the label given by the second argument (if set).

```
6199 \DeclareRobustCommand*{\glsnameaccessdisplay}[2]{%
6200   \protected@edef\@glo@access{\glsentryaccess{#2}}%
6201   \ifx\@glo@access\@gls@noaccess
6202     #1%
6203   \else
6204     \xglsaccsupp{\@glo@access}{#1}%
6205   \fi
6206 }
```

`\glsentrytextaccessdisplay` As above but for the `textaccess` replacement text.

```
6207 \DeclareRobustCommand*{\glsentrytextaccessdisplay}[2]{%
6208   \protected@edef\@glo@access{\glsentrytextaccess{#2}}%
6209   \ifx\@glo@access\@gls@noaccess
6210     #1%
6211   \else
6212     \xglsaccsupp{\@glo@access}{#1}%
6213   \fi
6214 }
```

`\glsentrypluralaccessdisplay` As above but for the `pluralaccess` replacement text.

```
6215 \DeclareRobustCommand*{\glsentrypluralaccessdisplay}[2]{%
6216   \protected@edef\@glo@access{\glsentrypluralaccess{#2}}%
6217   \ifx\@glo@access\@gls@noaccess
6218     #1%
6219   \else
6220     \xglsaccsupp{\@glo@access}{#1}%
6221   \fi
6222 }
```

`\glsentryfirstaccessdisplay` As above but for the `firstaccess` replacement text.

```
6223 \DeclareRobustCommand*{\glsentryfirstaccessdisplay}[2]{%
6224   \protected@edef\@glo@access{\glsentryfirstaccess{#2}}%
6225   \ifx\@glo@access\@gls@noaccess
6226     #1%
```

```

6227 \else
6228   \xglsaccsupp{\@glo@access}{#1}%
6229 \fi
6230 }

```

pluralaccessdisplay As above but for the firstpluralaccess replacement text.

```

6231 \DeclareRobustCommand*\glsfirstpluralaccessdisplay}[2]{%
6232   \protected@edef\@glo@access{\glsentryfirstpluralaccess{#2}}%
6233   \ifx\@glo@access\@gls@noaccess
6234     #1%
6235   \else
6236     \xglsaccsupp{\@glo@access}{#1}%
6237   \fi
6238 }

```

symbolaccessdisplay As above but for the symbolaccess replacement text.

```

6239 \DeclareRobustCommand*\glsymbolaccessdisplay}[2]{%
6240   \protected@edef\@glo@access{\glsentrysymbolaccess{#2}}%
6241   \ifx\@glo@access\@gls@noaccess
6242     #1%
6243   \else
6244     \xglsaccsupp{\@glo@access}{#1}%
6245   \fi
6246 }

```

pluralaccessdisplay As above but for the symbolpluralaccess replacement text.

```

6247 \DeclareRobustCommand*\glsymbolpluralaccessdisplay}[2]{%
6248   \protected@edef\@glo@access{\glsentrysymbolpluralaccess{#2}}%
6249   \ifx\@glo@access\@gls@noaccess
6250     #1%
6251   \else
6252     \xglsaccsupp{\@glo@access}{#1}%
6253   \fi
6254 }

```

descriptionaccessdisplay As above but for the descriptionaccess replacement text.

```

6255 \DeclareRobustCommand*\glsdescriptionaccessdisplay}[2]{%
6256   \protected@edef\@glo@access{\glsentrydescaccess{#2}}%
6257   \ifx\@glo@access\@gls@noaccess
6258     #1%
6259   \else
6260     \xglsaccsupp{\@glo@access}{#1}%
6261   \fi
6262 }

```

pluralaccessdisplay As above but for the descriptionpluralaccess replacement text.

```

6263 \DeclareRobustCommand*\glsdescriptionpluralaccessdisplay}[2]{%
6264   \protected@edef\@glo@access{\glsentrydescpluralaccess{#2}}%
6265   \ifx\@glo@access\@gls@noaccess

```

```

6266     #1%
6267   \else
6268     \xglsaccsupp{\@glo@access}{#1}%
6269   \fi
6270 }

```

`\glsshortaccessdisplay` As above but for the `shortaccess` replacement text.

```

6271 \DeclareRobustCommand*\glsshortaccessdisplay[2]{%
6272   \protected@edef\@glo@access{\glsentryshortaccess{#2}}%
6273   \ifx\@glo@access\@gls@noaccess
6274     #1%
6275   \else
6276     \xglsaccsupp{\@glo@access}{#1}%
6277   \fi
6278 }

```

`\glspluralaccessdisplay` As above but for the `shortpluralaccess` replacement text.

```

6279 \DeclareRobustCommand*\glsshortpluralaccessdisplay[2]{%
6280   \protected@edef\@glo@access{\glsentryshortpluralaccess{#2}}%
6281   \ifx\@glo@access\@gls@noaccess
6282     #1%
6283   \else
6284     \xglsaccsupp{\@glo@access}{#1}%
6285   \fi
6286 }

```

`\glslongaccessdisplay` As above but for the `longaccess` replacement text.

```

6287 \DeclareRobustCommand*\glslongaccessdisplay[2]{%
6288   \protected@edef\@glo@access{\glsentrylongaccess{#2}}%
6289   \ifx\@glo@access\@gls@noaccess
6290     #1%
6291   \else
6292     \xglsaccsupp{\@glo@access}{#1}%
6293   \fi
6294 }

```

`\glspluralaccessdisplay` As above but for the `longpluralaccess` replacement text.

```

6295 \DeclareRobustCommand*\glslongpluralaccessdisplay[2]{%
6296   \protected@edef\@glo@access{\glsentrylongpluralaccess{#2}}%
6297   \ifx\@glo@access\@gls@noaccess
6298     #1%
6299   \else
6300     \xglsaccsupp{\@glo@access}{#1}%
6301   \fi
6302 }

```

`\glsaccessdisplay` Gets the replacement text corresponding to the named key given by the first argument and calls the appropriate command defined above.

```

6303 \DeclareRobustCommand*\glsaccessdisplay}[3]{%
6304   \@ifundefined{gls#1accessdisplay}%
6305   {%
6306     \PackageError{glossaries-accsupp}{No accessibility support
6307       for key ‘#1’}{%
6308     }%
6309   }%
6310   \csname gls#1accessdisplay\endcsname{#2}{#3}%
6311 }%
6312 }

```

`\@gls@` Redefine `\@gls@` to change the way the link text is defined

```

6313 \def\@gls@#1#2[#3]{%
6314   \glsdoifexists{#2}%
6315   {%
6316     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

6317   \def\@gls@link@opts{#1}%
6318   \def\@gls@link@label{#2}%

```

Determine what the link text should be (this is stored in `\@glo@text`). This is no longer expanded.

```

6319   \ifglsused{#2}%
6320   {%
6321     \def\@glo@text{\csname gls@\@glo@type @display\endcsname
6322       {\glsfirstaccessdisplay{\glsentryfirst{#2}}{#2}}%
6323       {\glsdescriptionaccessdisplay{\glsentrydesc{#2}}{#2}}%
6324       {\glsymbolaccessdisplay{\glsentrysymbol{#2}}{#2}}%
6325       {#3}}%
6326   }%
6327   {%
6328     \def\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
6329       {\glsfirstaccessdisplay{\glsentryfirst{#2}}{#2}}%
6330       {\glsdescriptionaccessdisplay{\glsentrydesc{#2}}{#2}}%
6331       {\glsymbolaccessdisplay{\glsentrysymbol{#2}}{#2}}%
6332       {#3}}%
6333   }%

```

Call `\@gls@link`. If footnote package option has been used, suppress hyperlink for first use.

```

6334   \ifglsused{#2}%
6335   {%
6336     \@gls@link[#1]{#2}{\@glo@text}%
6337   }%
6338   {%
6339     \gls@checkisacronymlist\@glo@type
6340     \ifthenelse{\boolean{glsisacronymlist}\AND
6341       \boolean{glsacrfootnote}}{\OR\NOT\boolean{glshyperfirst}}%
6342   }%

```

```

6343      \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
6344      }%
6345      {%
6346      \@gls@link[#1]{#2}{\@glo@text}%
6347      }%
6348      }%

```

Indicate that this entry has now been used

```

6349      \glsunset{#2}%
6350      }%
6351      }

```

\@Gls@

```

6352 \def\@Gls@#1#2[#3]{%
6353   \glsdoifexists{#2}%
6354   {%
6355     \edef\@glo@type{\glentrytype{#2}}%

```

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

```

6356     \def\@gls@link@opts{#1}%
6357     \def\@gls@link@label{#2}%

```

Determine what the link text should be (this is stored in \@glo@text). The first character of the entry text is converted to uppercase before passing to \@gls@<type>@display or \@gls@<type>@displayfirst

```

6358     \ifglsused{#2}%
6359     {%
6360       \def\@glo@text{\csname gls@\@glo@type @display\endcsname
6361         {\glstextaccessdisplay{\Glsentrytext{#2}}{#2}}}%
6362       {\glsdescriptionaccessdisplay{\glentrydesc{#2}}{#2}}%
6363       {\glssymbolaccessdisplay{\glentrysymbol{#2}}{#2}}%
6364       {#3}}%
6365     }%
6366     {%
6367       \def\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
6368         {\glsfirstaccessdisplay{\Glsentryfirst{#2}}{#2}}%
6369         {\glsdescriptionaccessdisplay{\glentrydesc{#2}}{#2}}%
6370         {\glssymbolaccessdisplay{\glentrysymbol{#2}}{#2}}%
6371         {#3}}%
6372     }%

```

Call \@gls@link. If footnote package option has been used, suppress hyperlink for first use.

```

6373     \ifglsused{#2}%
6374     {%
6375       \@gls@link[#1]{#2}{\@glo@text}%
6376     }%
6377     {%
6378       \gls@checkisacronymlist\@glo@type
6379       \ifthenelse{\boolean{glsisacronymlist}}{\AND

```

```

6380     \boolean{glsacrfootnote}\) \OR\NOT\boolean{glshyperfirst}}}%
6381     {%
6382     \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
6383     }%
6384     {%
6385     \@gls@link[#1]{#2}{\@glo@text}%
6386     }%
6387     }%

```

Indicate that this entry has now been used

```

6388     \glsunset{#2}%
6389     }%
6390 }

```

\@GLS@

```

6391 \def\@GLS@#1#2[#3]{%
6392   \glsdoifexists{#2}{%
6393     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

6394   \def\@gls@link@opts{#1}%
6395   \def\@gls@link@label{#2}%

```

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

```

6396   \ifglsused{#2}%
6397   {%
6398     \def\@glo@text{\csname gls@\@glo@type @display\endcsname
6399       {\gls@textaccessdisplay{\glsentrytext{#2}}{#2}}}%
6400     {\gls@descriptionaccessdisplay{\glsentrydesc{#2}}{#2}}}%
6401     {\gls@symbolaccessdisplay{\glsentrysymbol{#2}}{#2}}}%
6402     {#3}}%
6403   }%
6404   {%
6405     \edef\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
6406       {\gls@firstaccessdisplay{\glsentryfirst{#2}}{#2}}}%
6407     {\gls@descriptionaccessdisplay{\glsentrydesc{#2}}{#2}}}%
6408     {\gls@symbolaccessdisplay{\glsentrysymbol{#2}}{#2}}}%
6409     {#3}}%
6410   }%

```

Call \@gls@link If footnote package option has been used, suppress hyperlink for first use.

```

6411   \ifglsused{#2}%
6412   {%
6413     \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}}%
6414   }%
6415   {%
6416     \gls@checkisacronymlist\@glo@type
6417     \ifthenelse{(\boolean{glsisacronymlist})\AND
6418       \boolean{glsacrfootnote}\) \OR\NOT\boolean{glshyperfirst}}{%
6419       \@gls@link[#1,hyper=false]{#2}{\MakeUppercase{\@glo@text}}}%

```

```

6420     }%
6421     {%
6422         \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
6423     }%
6424 }%

```

Indicate that this entry has now been used

```

6425     \glsunset{#2}%
6426 }%
6427 }

```

\@gls@pl@

```

6428 \def\@glspl@#1#2[#3]{%
6429     \glsdoifexists{#2}%
6430     {%
6431         \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

6432     \def\@gls@link@opts{#1}%
6433     \def\@gls@link@label{#2}%

```

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

```

6434     \ifglsused{#2}%
6435     {%
6436         \def\@glo@text{\csname gls@\@glo@type @display\endcsname
6437             {\glspluralaccessdisplay{\glsentryplural{#2}}{#2}}%
6438             {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
6439             {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
6440             {#3}}%
6441     }%
6442     {%
6443         \def\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
6444             {\glsfirstpluralaccessdisplay{\glsentryfirstplural{#2}}{#2}}%
6445             {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
6446             {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
6447             {#3}}%
6448     }%

```

Call \@gls@link If footnote package option has been used, suppress hyperlink for first use.

```

6449     \ifglsused{#2}%
6450     {%
6451         \@gls@link[#1]{#2}{\@glo@text}%
6452     }%
6453     {%
6454         \gls@checkisacronymlist\@glo@type
6455         \ifthenelse{(\boolean{@glsisacronymlist})\AND
6456             \boolean{glsacrfootnote}}{\OR\NOT\boolean{glshyperfirst}}%
6457         {%
6458             \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
6459         }%

```



```

6460      {%
6461      \@gls@link[#1]{#2}{\@glo@text}%
6462      }%
6463      }%

```

Indicate that this entry has now been used

```

6464      \glsunset{#2}%
6465      }%
6466 }

```

\@Glspl@

```

6467 \def\@Glspl@#1#2[#3]{%
6468   \glsdoifexists{#2}%
6469   {%
6470     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

6471     \def\@gls@link@opts{#1}%
6472     \def\@gls@link@label{#2}%

```

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

```

6473     \ifglsused{#2}%
6474     {%
6475       \def\@glo@text{\csname gls@\@glo@type @display\endcsname
6476         {\glspluralaccessdisplay{\Glsentryplural{#2}}{#2}}%
6477         {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
6478         {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
6479         {#3}}%
6480     }%
6481     {%
6482       \def\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
6483         {\glsfirstpluralaccessdisplay{\Glsentryfirstplural{#2}}{#2}}%
6484         {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
6485         {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
6486         {#3}}%
6487     }%

```

Call \@gls@link If footnote package option has been used, suppress hyperlink for first use.

```

6488     \ifglsused{#2}%
6489     {%
6490       \@gls@link[#1]{#2}{\@glo@text}%
6491     }%
6492     {%
6493       \ifthenelse{\equal{\@glo@type}{\acronymtype}\and
6494         \boolean{glsacrfootnote}}{%
6495         {%
6496           \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
6497         }%
6498       }%
6499       \@gls@link[#1]{#2}{\@glo@text}%

```

```

6500     }%
6501 }%

```

Indicate that this entry has now been used

```

6502     \glsunset{#2}%
6503 }%
6504 }

```

\@GLSp1@

```

6505 \def\@GLSp1@#1#2[#3]{%
6506   \glsdoifexists{#2}%
6507   {%
6508     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

6509     \def\@gls@link@opts{#1}%
6510     \def\@gls@link@label{#2}%

```

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

```

6511     \ifglsused{#2}%
6512     {%
6513       \def\@glo@text{\csname gls@\@glo@type @display\endcsname
6514         {\glspluralaccessdisplay{\glsentryplural{#2}}{#2}}%
6515         {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
6516         {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
6517         {#3}}%
6518     }%
6519     {%
6520       \def\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
6521         {\glsfirstpluralaccessdisplay{\glsentryfirstplural{#2}}{#2}}%
6522         {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
6523         {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
6524         {#3}}%
6525     }%

```

Call \@gls@link If footnote package option has been used, suppress hyperlink for first use.

```

6526     \ifglsused{#2}%
6527     {%
6528       \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
6529     }%
6530     {%
6531       \gls@checkisacronymlist\@glo@type
6532       \ifthenelse{(\boolean{glsisacronymlist}\AND
6533         \boolean{glsacrfootnote})\OR\not\boolean{glshyperfirst}}%
6534       {%
6535         \@gls@link[#1,hyper=false]{#2}{\MakeUppercase{\@glo@text}}%
6536       }%
6537       {%
6538         \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
6539       }%

```

```

6540 }%
        Indicate that this entry has now been used
6541 \glsunset{#2}%
6542 }%
6543 }

```

\@acrshort

```

6544 \def\@acrshort#1#2[#3]{%
6545   \glsdoifexists{#2}%
6546   {%
6547     \edef\@glo@type{\glsentrytype{#2}}%
        Determine what the link text should be (this is stored in \@glo@text)
6548     \def\@glo@text{%
6549       \glsshortaccessdisplay{\glsentryshort{#2}}{#2}%
6550     }%
        Call \@gls@link
6551     \@gls@link[#1]{#2}{\acronymfont{\@glo@text}#3}%
6552   }%
6553 }

```

\@Acrshort

```

6554 \def\@Acrshort#1#2[#3]{%
6555   \glsdoifexists{#2}%
6556   {%
6557     \edef\@glo@type{\glsentrytype{#2}}%
        Determine what the link text should be (this is stored in \@glo@text)
6558     \def\@glo@text{%
6559       \glsshortaccessdisplay{\Glsentryshort{#2}}{#2}%
6560     }%
        Call \@gls@link
6561     \@gls@link[#1]{#2}{\acronymfont{\@glo@text}#3}%
6562   }%
6563 }

```

\@ACRshort

```

6564 \def\@ACRshort#1#2[#3]{%
6565   \glsdoifexists{#2}%
6566   {%
6567     \edef\@glo@type{\glsentrytype{#2}}%
        Determine what the link text should be (this is stored in \@glo@text)
6568     \def\@glo@text{%
6569       \glsshortaccessdisplay{\MakeUppercase{\glsentryshort{#2}}}{#2}%
6570     }%

```

```

Call \@gls@link
6571   \@gls@link[#1]{#2}{\acronymfont{\@glo@text#3}}%
6572   }%
6573 }

```

\@acrlong

```

6574 \def\@acrlong#1#2[#3]{%
6575   \glsdoifexists{#2}%
6576   {%
6577     \edef\@glo@type{\glsentrytype{#2}}%
        Determine what the link text should be (this is stored in \@glo@text)
6578     \def\@glo@text{%
6579       \glslongaccessdisplay{\glsentrylong{#2}}{#2}%
6580     }%
        Call \@gls@link
6581     \@gls@link[#1]{#2}{\@glo@text#3}%
6582     }%
6583 }

```

\@Acrlong

```

6584 \def\@Acrlong#1#2[#3]{%
6585   \glsdoifexists{#2}%
6586   {%
6587     \edef\@glo@type{\glsentrytype{#2}}%
        Determine what the link text should be (this is stored in \@glo@text)
6588     \def\@glo@text{%
6589       \glslongaccessdisplay{\Glsentrylong{#2}}{#2}%
6590     }%
        Call \@gls@link
6591     \@gls@link[#1]{#2}{\@glo@text#3}%
6592     }%
6593 }

```

\@ACRlong

```

6594 \def\@ACRlong#1#2[#3]{%
6595   \glsdoifexists{#2}%
6596   {%
6597     \edef\@glo@type{\glsentrytype{#2}}%
        Determine what the link text should be (this is stored in \@glo@text)
6598     \def\@glo@text{%
6599       \glslongaccessdisplay{\MakeUppercase{\glsentrylong{#2}}}{#2}%
6600     }%
        Call \@gls@link
6601     \@gls@link[#1]{#2}{\@glo@text#3}%
6602     }%
6603 }

```

### 5.3 Displaying the Glossary

Entries within the glossary or list of acronyms are now formatted via `\accsuppglossaryentryfield` and `\accsuppglossarysubentryfield`.

`@glossaryentryfield`

```
6604 \ifglxindy
6605   \renewcommand*{\@glossaryentryfield}{%
6606     \string\accsuppglossaryentryfield}
6607 \else
6608   \renewcommand*{\@glossaryentryfield}{%
6609     \string\accsuppglossaryentryfield}
6610 \fi
```

`ossarysubentryfield`

```
6611 \ifglxindy
6612   \renewcommand*{\@glossarysubentryfield}{%
6613     \string\accsuppglossarysubentryfield}
6614 \else
6615   \renewcommand*{\@glossarysubentryfield}{%
6616     \string\accsuppglossarysubentryfield}
6617 \fi
```

`pglossaryentryfield`

```
6618 \newcommand*{\accsuppglossaryentryfield}[5]{%
6619   \glossaryentryfield{#1}%
6620   {\glxnameaccessdisplay{#2}{#1}}%
6621   {\glxdescriptionaccessdisplay{#3}{#1}}%
6622   {\glxsymbolaccessdisplay{#4}{#1}}{#5}%
6623 }
```

`ossarysubentryfield`

```
6624 \newcommand*{\accsuppglossarysubentryfield}[6]{%
6625   \glossaryentryfield{#1}{#2}%
6626   {\glxnameaccessdisplay{#3}{#2}}%
6627   {\glxdescriptionaccessdisplay{#4}{#2}}%
6628   {\glxsymbolaccessdisplay{#5}{#2}}{#6}%
6629 }
```

### 5.4 Acronyms

Use `\newacronymhook` to modify the key list to set the access text to the long version by default.

```
6630 \renewcommand*{\newacronymhook}{%
6631   \edef\@gls@keylist{shortaccess=\the\glxlongtok,%
6632     \the\glxkeylisttok}%
6633   \expandafter\glxkeylisttok\expandafter{\@gls@keylist}%
6634 }
```

efaultNewAcronymDef Modify default style to use access text:

```

6635 \renewcommand*{\DefaultNewAcronymDef}{%
6636   \edef\@do@newglossaryentry{%
6637     \noexpand\newglossaryentry{\the\glslabeltok}%
6638     {%
6639       type=\acronymtype,%
6640       name={\the\glsshorttok},%
6641       description={\the\glslongtok},%
6642       descriptionaccess=\relax,
6643       text={\the\glsshorttok},%
6644       access={\noexpand\@glo@textaccess},%
6645       sort={\the\glsshorttok},%
6646       short={\the\glsshorttok},%
6647       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6648       shortaccess={\the\glslongtok},%
6649       long={\the\glslongtok},%
6650       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6651       descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6652       first={\noexpand\glslongaccessdisplay
6653         {\the\glslongtok}{\the\glslabeltok}\space
6654         (\noexpand\glsshortaccessdisplay
6655           {\the\glsshorttok}{\the\glslabeltok})},%
6656       plural={\the\glsshorttok\acrpluralsuffix},%
6657       firstplural={\noexpand\glslongpluralaccessdisplay
6658         {\noexpand\@glo@longpl}{\the\glslabeltok}\space
6659         (\noexpand\glsshortpluralaccessdisplay
6660           {\noexpand\@glo@shortpl}{\the\glslabeltok})},%
6661       firstaccess=\relax,
6662       firstpluralaccess=\relax,
6663       textaccess={\noexpand\@glo@shortaccess},%
6664       \the\glskeylisttok
6665     }%
6666   }%
6667   \@do@newglossaryentry
6668 }

```

otnoteNewAcronymDef

```

6669 \renewcommand*{\DescriptionFootnoteNewAcronymDef}{%
6670   \edef\@do@newglossaryentry{%
6671     \noexpand\newglossaryentry{\the\glslabeltok}%
6672     {%
6673       type=\acronymtype,%
6674       name={\noexpand\acronymfont{\the\glsshorttok}},%
6675       sort={\the\glsshorttok},%
6676       text={\the\glsshorttok},%
6677       short={\the\glsshorttok},%
6678       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6679       shortaccess={\the\glslongtok},%
6680       long={\the\glslongtok},%

```

```

6681     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6682     access={\noexpand\@glo@textaccess},%
6683     plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6684     symbol={\the\glslongtok},%
6685     symbolplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6686     firstpluralaccess=\relax,
6687     textaccess={\noexpand\@glo@shortaccess},%
6688     \the\glskeylisttok
6689   }%
6690 }%
6691 \@do@newglossaryentry
6692 }

```

ptionNewAcronymDef

```

6693 \renewcommand*{\DescriptionNewAcronymDef}{%
6694   \edef\@do@newglossaryentry{%
6695     \noexpand\newglossaryentry{\the\glslabeltok}%
6696     {%
6697       type=\acronymtype,%
6698       name={\noexpand
6699         \acrnameformat{\the\glsshorttok}{\the\glslongtok}},%
6700       access={\noexpand\@glo@textaccess},%
6701       sort={\the\glsshorttok},%
6702       short={\the\glsshorttok},%
6703       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6704       shortaccess={\the\glslongtok},%
6705       long={\the\glslongtok},%
6706       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6707       first={\the\glslongtok},%
6708       firstaccess=\relax,
6709       firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6710       text={\the\glsshorttok},%
6711       textaccess={\the\glslongtok},%
6712       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6713       symbol={\noexpand\@glo@text},%
6714       symbolaccess={\noexpand\@glo@textaccess},%
6715       symbolplural={\noexpand\@glo@plural},%
6716       firstpluralaccess=\relax,
6717       textaccess={\noexpand\@glo@shortaccess},%
6718       \the\glskeylisttok}%
6719   }%
6720   \@do@newglossaryentry
6721 }

```

otnoteNewAcronymDef

```

6722 \renewcommand*{\FootnoteNewAcronymDef}{%
6723   \edef\@do@newglossaryentry{%
6724     \noexpand\newglossaryentry{\the\glslabeltok}%
6725     {%

```

```

6726     type=\acronymtype,%
6727     name={\noexpand\acronymfont{\the\glsshorttok}},%
6728     sort={\the\glsshorttok},%
6729     text={\the\glsshorttok},%
6730     textaccess={\the\glslongtok},%
6731     access={\noexpand\@glo@textaccess},%
6732     plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6733     short={\the\glsshorttok},%
6734     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6735     long={\the\glslongtok},%
6736     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6737     description={\the\glslongtok},%
6738     descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6739     \the\glskeylisttok
6740 }%
6741 }%
6742 \@do@newglossaryentry
6743 }

```

\SmallNewAcronymDef

```

6744 \renewcommand*{\SmallNewAcronymDef}{%
6745   \edef\@do@newglossaryentry{%
6746     \noexpand\newglossaryentry{\the\glslabeltok}%
6747     {%
6748       type=\acronymtype,%
6749       name={\noexpand\acronymfont{\the\glsshorttok}},%
6750       access={\noexpand\@glo@symbolaccess},%
6751       sort={\the\glsshorttok},%
6752       short={\the\glsshorttok},%
6753       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6754       shortaccess={\the\glslongtok},%
6755       long={\the\glslongtok},%
6756       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6757       text={\noexpand\@glo@short},%
6758       textaccess={\noexpand\@glo@shortaccess},%
6759       plural={\noexpand\@glo@shortpl},%
6760       first={\the\glslongtok},%
6761       firstaccess=\relax,
6762       firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6763       description={\noexpand\@glo@first},%
6764       descriptionplural={\noexpand\@glo@firstplural},%
6765       symbol={\the\glsshorttok},%
6766       symbolaccess={\the\glslongtok},%
6767       symbolplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6768       \the\glskeylisttok
6769     }%
6770   }%
6771   \@do@newglossaryentry
6772 }

```



The following are kept for compatibility with versions before 3.0:

```
\glsshortaccesskey
6773 \newcommand*{\glsshortaccesskey}{\glsshortkey access}%

\shortpluralaccesskey
6774 \newcommand*{\glsshortpluralaccesskey}{\glsshortpluralkey access}%

\glslongaccesskey
6775 \newcommand*{\glslongaccesskey}{\glslongkey access}%

\longpluralaccesskey
6776 \newcommand*{\glslongpluralaccesskey}{\glslongpluralkey access}%
```

## 5.5 Debugging Commands

```
\showglonameaccess
6777 \newcommand*{\showglonameaccess}[1]{%
6778 \expandafter\show\csname glo@#1@textaccess\endcsname
6779 }

\showglotextaccess
6780 \newcommand*{\showglotextaccess}[1]{%
6781 \expandafter\show\csname glo@#1@textaccess\endcsname
6782 }

\showglopluralaccess
6783 \newcommand*{\showglopluralaccess}[1]{%
6784 \expandafter\show\csname glo@#1@pluralaccess\endcsname
6785 }

\showglofirstaccess
6786 \newcommand*{\showglofirstaccess}[1]{%
6787 \expandafter\show\csname glo@#1@firstaccess\endcsname
6788 }

\glofirstpluralaccess
6789 \newcommand*{\showglofirstpluralaccess}[1]{%
6790 \expandafter\show\csname glo@#1@firstpluralaccess\endcsname
6791 }

\showglosymbolaccess
6792 \newcommand*{\showglosymbolaccess}[1]{%
6793 \expandafter\show\csname glo@#1@symbolaccess\endcsname
6794 }
```

osymbolpluralaccess

```
6795 \newcommand*{\showglosymbolpluralaccess}[1]{%
6796   \expandafter\show\csname glo@#1@symbolpluralaccess\endcsname
6797 }
```

\showglodescaccess

```
6798 \newcommand*{\showglodescaccess}[1]{%
6799   \expandafter\show\csname glo@#1@descaccess\endcsname
6800 }
```

glodescpluralaccess

```
6801 \newcommand*{\showglodescpluralaccess}[1]{%
6802   \expandafter\show\csname glo@#1@descpluralaccess\endcsname
6803 }
```

\showgloshortaccess

```
6804 \newcommand*{\showgloshortaccess}[1]{%
6805   \expandafter\show\csname glo@#1@shortaccess\endcsname
6806 }
```

loshortpluralaccess

```
6807 \newcommand*{\showgloshortpluralaccess}[1]{%
6808   \expandafter\show\csname glo@#1@shortpluralaccess\endcsname
6809 }
```

\showglolongaccess

```
6810 \newcommand*{\showglolongaccess}[1]{%
6811   \expandafter\show\csname glo@#1@longaccess\endcsname
6812 }
```

glolongpluralaccess

```
6813 \newcommand*{\showglolongpluralaccess}[1]{%
6814   \expandafter\show\csname glo@#1@longpluralaccess\endcsname
6815 }
```

## 6 Multi-Lingual Support

Many thanks to everyone who contributed to the translations both via email and on comp.text.tex.

### 6.1 Babel Captions

Define captions if multi-lingual support is required, but the package is not loaded.

```
6816 \NeedsTeXFormat{LaTeX2e}
6817 \ProvidesPackage{glossaries-babel}[2009/04/16 v1.2 (NLCT)]
```

English:

```
6818 \@ifundefined{captionsenglish}{%}{%
6819   \addto\captionsenglish{%
6820     \renewcommand*{\glossaryname}{Glossary}%
6821     \renewcommand*{\acronymname}{Acronyms}%
6822     \renewcommand*{\entryname}{Notation}%
6823     \renewcommand*{\descriptionname}{Description}%
6824     \renewcommand*{\symbolname}{Symbol}%
6825     \renewcommand*{\pagelistname}{Page List}%
6826     \renewcommand*{\glssymbolsgroupname}{Symbols}%
6827     \renewcommand*{\glsnumbersgroupname}{Numbers}%
6828 }%
6829 }
6830 \@ifundefined{captionsamerican}{%}{%
6831   \addto\captionsamerican{%
6832     \renewcommand*{\glossaryname}{Glossary}%
6833     \renewcommand*{\acronymname}{Acronyms}%
6834     \renewcommand*{\entryname}{Notation}%
6835     \renewcommand*{\descriptionname}{Description}%
6836     \renewcommand*{\symbolname}{Symbol}%
6837     \renewcommand*{\pagelistname}{Page List}%
6838     \renewcommand*{\glssymbolsgroupname}{Symbols}%
6839     \renewcommand*{\glsnumbersgroupname}{Numbers}%
6840 }%
6841 }
6842 \@ifundefined{captionsaustralian}{%}{%
6843   \addto\captionsaustralian{%
6844     \renewcommand*{\glossaryname}{Glossary}%
6845     \renewcommand*{\acronymname}{Acronyms}%
6846     \renewcommand*{\entryname}{Notation}%
6847     \renewcommand*{\descriptionname}{Description}%
6848     \renewcommand*{\symbolname}{Symbol}%
6849     \renewcommand*{\pagelistname}{Page List}%
6850     \renewcommand*{\glssymbolsgroupname}{Symbols}%
6851     \renewcommand*{\glsnumbersgroupname}{Numbers}%
6852 }%
6853 }
6854 \@ifundefined{captionsbritish}{%}{%
6855   \addto\captionsbritish{%
6856     \renewcommand*{\glossaryname}{Glossary}%
6857     \renewcommand*{\acronymname}{Acronyms}%
6858     \renewcommand*{\entryname}{Notation}%
6859     \renewcommand*{\descriptionname}{Description}%
6860     \renewcommand*{\symbolname}{Symbol}%
6861     \renewcommand*{\pagelistname}{Page List}%
6862     \renewcommand*{\glssymbolsgroupname}{Symbols}%
6863     \renewcommand*{\glsnumbersgroupname}{Numbers}%
6864 }}%
6865 \@ifundefined{captionscanadian}{%}{%
```

```

6866 \addto\captionscanadian{%
6867 \renewcommand*{\glossaryname}{Glossary}%
6868 \renewcommand*{\acronymname}{Acronyms}%
6869 \renewcommand*{\entryname}{Notation}%
6870 \renewcommand*{\descriptionname}{Description}%
6871 \renewcommand*{\symbolname}{Symbol}%
6872 \renewcommand*{\pagelistname}{Page List}%
6873 \renewcommand*{\glssymbolsgroupname}{Symbols}%
6874 \renewcommand*{\glsnumbersgroupname}{Numbers}%
6875 }%
6876 }
6877 \@ifundefined{captionsnewzealand}{}{%
6878 \addto\captionscanadian{%
6879 \renewcommand*{\glossaryname}{Glossary}%
6880 \renewcommand*{\acronymname}{Acronyms}%
6881 \renewcommand*{\entryname}{Notation}%
6882 \renewcommand*{\descriptionname}{Description}%
6883 \renewcommand*{\symbolname}{Symbol}%
6884 \renewcommand*{\pagelistname}{Page List}%
6885 \renewcommand*{\glssymbolsgroupname}{Symbols}%
6886 \renewcommand*{\glsnumbersgroupname}{Numbers}%
6887 }%
6888 }
6889 \@ifundefined{captionsUKenglish}{}{%
6890 \addto\captionscanadian{%
6891 \renewcommand*{\glossaryname}{Glossary}%
6892 \renewcommand*{\acronymname}{Acronyms}%
6893 \renewcommand*{\entryname}{Notation}%
6894 \renewcommand*{\descriptionname}{Description}%
6895 \renewcommand*{\symbolname}{Symbol}%
6896 \renewcommand*{\pagelistname}{Page List}%
6897 \renewcommand*{\glssymbolsgroupname}{Symbols}%
6898 \renewcommand*{\glsnumbersgroupname}{Numbers}%
6899 }%
6900 }
6901 \@ifundefined{captionsUSenglish}{}{%
6902 \addto\captionscanadian{%
6903 \renewcommand*{\glossaryname}{Glossary}%
6904 \renewcommand*{\acronymname}{Acronyms}%
6905 \renewcommand*{\entryname}{Notation}%
6906 \renewcommand*{\descriptionname}{Description}%
6907 \renewcommand*{\symbolname}{Symbol}%
6908 \renewcommand*{\pagelistname}{Page List}%
6909 \renewcommand*{\glssymbolsgroupname}{Symbols}%
6910 \renewcommand*{\glsnumbersgroupname}{Numbers}%
6911 }%
6912 }

```

German (quite a few variations were suggested for German; I settled on the following):

```

6913 \@ifundefined{captionsgerman}{\}%
6914   \addto\captionsgerman{%
6915     \renewcommand*{\glossaryname}{Glossar}%
6916     \renewcommand*{\acronymname}{Akronyme}%
6917     \renewcommand*{\entryname}{Bezeichnung}%
6918     \renewcommand*{\descriptionname}{Beschreibung}%
6919     \renewcommand*{\symbolname}{Symbol}%
6920     \renewcommand*{\pagelistname}{Seiten}%
6921     \renewcommand*{\glssymbolsgroupname}{Symbole}%
6922     \renewcommand*{\glsnumbersgroupname}{Zahlen}}
6923 }

```

ngerman is identical to German:

```

6924 \@ifundefined{captionsgerman}{\}%
6925   \addto\captionsgerman{%
6926     \renewcommand*{\glossaryname}{Glossar}%
6927     \renewcommand*{\acronymname}{Akronyme}%
6928     \renewcommand*{\entryname}{Bezeichnung}%
6929     \renewcommand*{\descriptionname}{Beschreibung}%
6930     \renewcommand*{\symbolname}{Symbol}%
6931     \renewcommand*{\pagelistname}{Seiten}%
6932     \renewcommand*{\glssymbolsgroupname}{Symbole}%
6933     \renewcommand*{\glsnumbersgroupname}{Zahlen}}
6934 }

```

Italian:

```

6935 \@ifundefined{captionssitalian}{\}%
6936   \addto\captionssitalian{%
6937     \renewcommand*{\glossaryname}{Glossario}%
6938     \renewcommand*{\acronymname}{Acronimi}%
6939     \renewcommand*{\entryname}{Nomenclatura}%
6940     \renewcommand*{\descriptionname}{Descrizione}%
6941     \renewcommand*{\symbolname}{Simbolo}%
6942     \renewcommand*{\pagelistname}{Elenco delle pagine}%
6943     \renewcommand*{\glssymbolsgroupname}{Simboli}%
6944     \renewcommand*{\glsnumbersgroupname}{Numeri}}
6945 }

```

Dutch:

```

6946 \@ifundefined{captionsdutch}{\}%
6947   \addto\captionsdutch{%
6948     \renewcommand*{\glossaryname}{Woordenlijst}%
6949     \renewcommand*{\acronymname}{Acroniemen}%
6950     \renewcommand*{\entryname}{Benaming}%
6951     \renewcommand*{\descriptionname}{Beschrijving}%
6952     \renewcommand*{\symbolname}{Symbool}%
6953     \renewcommand*{\pagelistname}{Pagina's}%
6954     \renewcommand*{\glssymbolsgroupname}{Symbolen}%

```

```

6955 \renewcommand*{\glsnumbersgroupname}{Cijfers}}
6956 }

Spanish:

6957 \@ifundefined{captionsspanish}{}{%
6958 \addto\captionsspanish{%
6959 \renewcommand*{\glossaryname}{Glosario}%
6960 \renewcommand*{\acronymname}{Siglas}%
6961 \renewcommand*{\entryname}{Entrada}%
6962 \renewcommand*{\descriptionname}{Descripci\'on}%
6963 \renewcommand*{\symbolname}{S\'{i}mbolo}%
6964 \renewcommand*{\pagelistname}{Lista de p\'aginas}%
6965 \renewcommand*{\glssymbolsgroupname}{S\'{i}mbolos}%
6966 \renewcommand*{\glsnumbersgroupname}{N\'umeros}}
6967 }

French:

6968 \@ifundefined{captionsfrench}{}{%
6969 \addto\captionsfrench{%
6970 \renewcommand*{\glossaryname}{Glossaire}%
6971 \renewcommand*{\acronymname}{Acronymes}%
6972 \renewcommand*{\entryname}{Terme}%
6973 \renewcommand*{\descriptionname}{Description}%
6974 \renewcommand*{\symbolname}{Symbole}%
6975 \renewcommand*{\pagelistname}{Pages}%
6976 \renewcommand*{\glssymbolsgroupname}{Symboles}%
6977 \renewcommand*{\glsnumbersgroupname}{Nombres}}
6978 }

6979 \@ifundefined{captionsfrenchb}{}{%
6980 \addto\captionsfrenchb{%
6981 \renewcommand*{\glossaryname}{Glossaire}%
6982 \renewcommand*{\acronymname}{Acronymes}%
6983 \renewcommand*{\entryname}{Terme}%
6984 \renewcommand*{\descriptionname}{Description}%
6985 \renewcommand*{\symbolname}{Symbole}%
6986 \renewcommand*{\pagelistname}{Pages}%
6987 \renewcommand*{\glssymbolsgroupname}{Symboles}%
6988 \renewcommand*{\glsnumbersgroupname}{Nombres}}
6989 }

6990 \@ifundefined{captionsfrançais}{}{%
6991 \addto\captionsfrançais{%
6992 \renewcommand*{\glossaryname}{Glossaire}%
6993 \renewcommand*{\acronymname}{Acronymes}%
6994 \renewcommand*{\entryname}{Terme}%
6995 \renewcommand*{\descriptionname}{Description}%
6996 \renewcommand*{\symbolname}{Symbole}%
6997 \renewcommand*{\pagelistname}{Pages}%
6998 \renewcommand*{\glssymbolsgroupname}{Symboles}%
6999 \renewcommand*{\glsnumbersgroupname}{Nombres}}
7000 }

```

Danish:

```
7001 \@ifundefined{captionsdanish}{%{}%
7002   \addto\captionsdanish{%
7003     \renewcommand*{\glossaryname}{Ordliste}%
7004     \renewcommand*{\acronymname}{Akronymer}%
7005     \renewcommand*{\entryname}{Symbolforklaring}%
7006     \renewcommand*{\descriptionname}{Beskrivelse}%
7007     \renewcommand*{\symbolname}{Symbol}%
7008     \renewcommand*{\pagelistname}{Side}%
7009     \renewcommand*{\glssymbolsgroupname}{Symboler}%
7010     \renewcommand*{\glsnumbersgroupname}{Tal}}
7011 }
```

Irish:

```
7012 \@ifundefined{captionsirish}{%{}%
7013   \addto\captionsirish{%
7014     \renewcommand*{\glossaryname}{Gluais}%
7015     \renewcommand*{\acronymname}{Acrainmneacha}%

```

wasn't sure whether to go for Nóta (Note), Ciall ('Meaning', 'sense') or Brí ('Meaning'). In the end I chose Ciall.

```
7016     \renewcommand*{\entryname}{Ciall}%
7017     \renewcommand*{\descriptionname}{Tuairisc}%

```

Again, not sure whether to use Comhartha/Comharthaí or Siombail/Siombaile, so have chosen the former.

```
7018     \renewcommand*{\symbolname}{Comhartha}%
7019     \renewcommand*{\glssymbolsgroupname}{Comhartha\'}{\i}}%
7020     \renewcommand*{\pagelistname}{Leathanaigh}%
7021     \renewcommand*{\glsnumbersgroupname}{Uimhreacha}}
7022 }
```

Hungarian:

```
7023 \@ifundefined{captionsmagyar}{%{}%
7024   \addto\captionsmagyar{%
7025     \renewcommand*{\glossaryname}{Sz\'ojegyz\'ek}%
7026     \renewcommand*{\acronymname}{Bet\H uszavak}%
7027     \renewcommand*{\entryname}{Kifejez\'es}%
7028     \renewcommand*{\descriptionname}{Magyar\'azat}%
7029     \renewcommand*{\symbolname}{Jel\'ol\'es}%
7030     \renewcommand*{\pagelistname}{Oldalsz\'am}%
7031     \renewcommand*{\glssymbolsgroupname}{Jelek}%
7032     \renewcommand*{\glsnumbersgroupname}{Sz\'amjegyek}%
7033   }
7034 }
7035 \@ifundefined{captionshungarian}{%{}%
7036   \addto\captionshungarian{%
7037     \renewcommand*{\glossaryname}{Sz\'ojegyz\'ek}%
7038     \renewcommand*{\acronymname}{Bet\H uszavak}%
7039     \renewcommand*{\entryname}{Kifejez\'es}%
7040     \renewcommand*{\descriptionname}{Magyar\'azat}%

```

```

7041 \renewcommand*{\symbolname}{Jel\`ol\`es}%
7042 \renewcommand*{\pagelistname}{Oldalsz\`am}%
7043 \renewcommand*{\glssymbolsgroupname}{Jelek}%
7044 \renewcommand*{\glsnumbersgroupname}{Sz\`amjegyek}%
7045 }
7046 }

```

#### Polish

```

7047 \@ifundefined{captionspolish}{}{%
7048 \addto\captionspolish{%
7049 \renewcommand*{\glossaryname}{S{\l}ownik termin\`ow}%
7050 \renewcommand*{\acronymname}{Skr\`ot}%
7051 \renewcommand*{\entryname}{Termin}%
7052 \renewcommand*{\descriptionname}{Opis}%
7053 \renewcommand*{\symbolname}{Symbol}%
7054 \renewcommand*{\pagelistname}{Strony}%
7055 \renewcommand*{\glssymbolsgroupname}{Symbole}%
7056 \renewcommand*{\glsnumbersgroupname}{Liczby}}
7057 }

```

#### Brazilian

```

7058 \@ifundefined{captionsbrazil}{}{%
7059 \addto\captionsbrazil{%
7060 \renewcommand*{\glossaryname}{Gloss\`ario}%
7061 \renewcommand*{\acronymname}{Siglas}%
7062 \renewcommand*{\entryname}{Nota\`c c\`ao}%
7063 \renewcommand*{\descriptionname}{Descri\`c c\`ao}%
7064 \renewcommand*{\symbolname}{S\`imbolo}%
7065 \renewcommand*{\pagelistname}{Lista de P\`aginas}%
7066 \renewcommand*{\glssymbolsgroupname}{S\`imbolos}%
7067 \renewcommand*{\glsnumbersgroupname}{N\`umeros}%
7068 }%
7069 }

```

## 6.2 Polyglossia Captions

```

7070 \NeedsTeXFormat{LaTeX2e}
7071 \ProvidesPackage{glossaries-polyglossia}[2009/11/09 v1.0 (NLCT)]

```

#### English:

```

7072 \@ifundefined{captionseenglish}{}{%
7073 \expandafter\toks@\expandafter{\captionseenglish
7074 \renewcommand*{\glossaryname}{\textenglish{Glossary}}%
7075 \renewcommand*{\acronymname}{\textenglish{Acronyms}}%
7076 \renewcommand*{\entryname}{\textenglish{Notation}}%
7077 \renewcommand*{\descriptionname}{\textenglish{Description}}%
7078 \renewcommand*{\symbolname}{\textenglish{Symbol}}%
7079 \renewcommand*{\pagelistname}{\textenglish{Page List}}%
7080 \renewcommand*{\glssymbolsgroupname}{\textenglish{Symbols}}%
7081 \renewcommand*{\glsnumbersgroupname}{\textenglish{Numbers}}%
7082 }%

```



```

7083 \edef\captionseenglish{\the\toks@}%
7084 }

```

#### German:

```

7085 \@ifundefined{captionsgerman}{}{%
7086 \expandafter\toks@\expandafter{\captionsgerman
7087 \renewcommand*{\glossaryname}{\textgerman{Glossar}}}%
7088 \renewcommand*{\acronymname}{\textgerman{Akronyme}}}%
7089 \renewcommand*{\entryname}{\textgerman{Bezeichnung}}}%
7090 \renewcommand*{\descriptionname}{\textgerman{Beschreibung}}}%
7091 \renewcommand*{\symbolname}{\textgerman{Symbol}}}%
7092 \renewcommand*{\pagelistname}{\textgerman{Seiten}}}%
7093 \renewcommand*{\glssymbolsgroupname}{\textgerman{Symbole}}}%
7094 \renewcommand*{\glsnumbersgroupname}{\textgerman{Zahlen}}}%
7095 }%
7096 \edef\captionsgerman{\the\toks@}%
7097 }

```

#### Italian:

```

7098 \@ifundefined{captionssitalian}{}{%
7099 \expandafter\toks@\expandafter{\captionssitalian
7100 \renewcommand*{\glossaryname}{\textitalian{Glossario}}}%
7101 \renewcommand*{\acronymname}{\textitalian{Acronimi}}}%
7102 \renewcommand*{\entryname}{\textitalian{Nomenclatura}}}%
7103 \renewcommand*{\descriptionname}{\textitalian{Descrizione}}}%
7104 \renewcommand*{\symbolname}{\textitalian{Simbolo}}}%
7105 \renewcommand*{\pagelistname}{\textitalian{Elenco delle pagine}}}%
7106 \renewcommand*{\glssymbolsgroupname}{\textitalian{Simboli}}}%
7107 \renewcommand*{\glsnumbersgroupname}{\textitalian{Numeri}}}%
7108 }%
7109 \edef\captionssitalian{\the\toks@}%
7110 }

```

#### Dutch:

```

7111 \@ifundefined{captionsdutch}{}{%
7112 \expandafter\toks@\expandafter{\captionsdutch
7113 \renewcommand*{\glossaryname}{\textdutch{Woordenlijst}}}%
7114 \renewcommand*{\acronymname}{\textdutch{Acroniemen}}}%
7115 \renewcommand*{\entryname}{\textdutch{Benaming}}}%
7116 \renewcommand*{\descriptionname}{\textdutch{Beschrijving}}}%
7117 \renewcommand*{\symbolname}{\textdutch{Symbool}}}%
7118 \renewcommand*{\pagelistname}{\textdutch{Pagina's}}}%
7119 \renewcommand*{\glssymbolsgroupname}{\textdutch{Symbolen}}}%
7120 \renewcommand*{\glsnumbersgroupname}{\textdutch{Cijfers}}}%
7121 }%
7122 \edef\captionsdutch{\the\toks@}%
7123 }

```

#### Spanish:

```

7124 \@ifundefined{captionssspanish}{}{%
7125 \expandafter\toks@\expandafter{\captionssspanish
7126 \renewcommand*{\glossaryname}{\textspanish{Glosario}}}%

```

```

7127 \renewcommand*{\acronymname}{\textspanish{Siglas}}%
7128 \renewcommand*{\entryname}{\textspanish{Entrada}}%
7129 \renewcommand*{\descriptionname}{\textspanish{Descripci' on}}%
7130 \renewcommand*{\symbolname}{\textspanish{S\'\i mbolo}}%
7131 \renewcommand*{\pagelistname}{\textspanish{Lista de p\ ' aginas}}%
7132 \renewcommand*{\glssymbolsgroupname}{\textspanish{S\'\i mbolos}}%
7133 \renewcommand*{\glslnumbersgroupname}{\textspanish{N\ ' umeros}}%
7134 }%
7135 \edef\captionsspanish{\the\toks@}%
7136 }

```

#### French:

```

7137 \@ifundefined{captionsfrench}{}{%
7138 \expandafter\toks@\expandafter{\captionsfrench
7139 \renewcommand*{\glossaryname}{\textfrench{Glossaire}}%
7140 \renewcommand*{\acronymname}{\textfrench{Acronymes}}%
7141 \renewcommand*{\entryname}{\textfrench{Terme}}%
7142 \renewcommand*{\descriptionname}{\textfrench{Description}}%
7143 \renewcommand*{\symbolname}{\textfrench{Symbole}}%
7144 \renewcommand*{\pagelistname}{\textfrench{Pages}}%
7145 \renewcommand*{\glssymbolsgroupname}{\textfrench{Symboles}}%
7146 \renewcommand*{\glslnumbersgroupname}{\textfrench{Nombres}}%
7147 }%
7148 \edef\captionsfrench{\the\toks@}%
7149 }

```

#### Danish:

```

7150 \@ifundefined{captionsdanish}{}{%
7151 \expandafter\toks@\expandafter{\captionsdanish
7152 \renewcommand*{\glossaryname}{\textdanish{Ordliste}}%
7153 \renewcommand*{\acronymname}{\textdanish{Akronymer}}%
7154 \renewcommand*{\entryname}{\textdanish{Symbolforklaring}}%
7155 \renewcommand*{\descriptionname}{\textdanish{Beskrivelse}}%
7156 \renewcommand*{\symbolname}{\textdanish{Symbol}}%
7157 \renewcommand*{\pagelistname}{\textdanish{Side}}%
7158 \renewcommand*{\glssymbolsgroupname}{\textdanish{Symboler}}%
7159 \renewcommand*{\glslnumbersgroupname}{\textdanish{Tal}}%
7160 }%
7161 \edef\captionsdanish{\the\toks@}%
7162 }

```

#### Irish:

```

7163 \@ifundefined{captionsirish}{}{%
7164 \expandafter\toks@\expandafter{\captionsirish
7165 \renewcommand*{\glossaryname}{\textirish{Gluais}}%
7166 \renewcommand*{\acronymname}{\textirish{Acrainmneacha}}%
7167 \renewcommand*{\entryname}{\textirish{Ciall}}%
7168 \renewcommand*{\descriptionname}{\textirish{Tuaireasc}}%
7169 \renewcommand*{\symbolname}{\textirish{Comhartha}}%
7170 \renewcommand*{\glssymbolsgroupname}{\textirish{Comhartha\ ' \i}}%
7171 \renewcommand*{\pagelistname}{\textirish{Leathanaigh}}%

```

```

7172 \renewcommand*{\glnumbersgroupname}{\textirish{Uimhreacha}}}%
7173 }%
7174 \edef\captionisirish{\the\toks@}%
7175 }

```

#### Hungarian:

```

7176 \@ifundefined{captionsmagyar}{\}%
7177 \expandafter\toks@\expandafter{\captionsmagyar
7178 \renewcommand*{\glossaryname}{\textmagyar{Sz\'ojegyz\'ek}}}%
7179 \renewcommand*{\acronymname}{\textmagyar{Bet\H uszavak}}}%
7180 \renewcommand*{\entryname}{\textmagyar{Kifejez\'es}}}%
7181 \renewcommand*{\descriptionname}{\textmagyar{Magyar\'azat}}}%
7182 \renewcommand*{\symbolname}{\textmagyar{Jel\'ol\'es}}}%
7183 \renewcommand*{\pagelistname}{\textmagyar{Oldalsz\'am}}}%
7184 \renewcommand*{\glssymbolsgroupname}{\textmagyar{Jelek}}}%
7185 \renewcommand*{\glnumbersgroupname}{\textmagyar{Sz\'amjegyek}}}%
7186 }%
7187 \edef\captionsmagyar{\the\toks@}%
7188 }

```

#### Polish

```

7189 \@ifundefined{captionspolish}{\}%
7190 \expandafter\toks@\expandafter{\captionspolish
7191 \renewcommand*{\glossaryname}{\textpolish{S\l ownnik termin\'ow}}}%
7192 \renewcommand*{\acronymname}{\textpolish{Skr\'ot}}}%
7193 \renewcommand*{\entryname}{\textpolish{Termin}}}%
7194 \renewcommand*{\descriptionname}{\textpolish{Opis}}}%
7195 \renewcommand*{\symbolname}{\textpolish{Symbol}}}%
7196 \renewcommand*{\pagelistname}{\textpolish{Strony}}}%
7197 \renewcommand*{\glssymbolsgroupname}{\textpolish{Symbole}}}%
7198 \renewcommand*{\glnumbersgroupname}{\textpolish{Liczby}}}%
7199 }%
7200 \edef\captionspolish{\the\toks@}%
7201 }

```

#### Portugues

```

7202 \@ifundefined{captionsportuges}{\}%
7203 \expandafter\toks@\expandafter{\captionsportuges
7204 \renewcommand*{\glossaryname}{\textportuges{Gloss\'ario}}}%
7205 \renewcommand*{\acronymname}{\textportuges{Siglas}}}%
7206 \renewcommand*{\entryname}{\textportuges{Nota\c c\~ao}}}%
7207 \renewcommand*{\descriptionname}{\textportuges{Descri\c c\~ao}}}%
7208 \renewcommand*{\symbolname}{\textportuges{S\'imbolo}}}%
7209 \renewcommand*{\pagelistname}{\textportuges{Lista de P\'aginas}}}%
7210 \renewcommand*{\glssymbolsgroupname}{\textportuges{S\'imbolos}}}%
7211 \renewcommand*{\glnumbersgroupname}{\textportuges{N\'umeros}}}%
7212 }%
7213 \edef\captionsportuges{\the\toks@}%
7214 }

```

### 6.3 Brazilian Dictionary

This is a dictionary file provided by Thiago de Melo for use with the package.

```
7215 \ProvidesDictionary{glossaries-dictionary}{Brazilian}
```

Provide Brazilian translations:

```
7216 \providetranslation{Glossary}{Gloss\`ario}
7217 \providetranslation{Acronyms}{Siglas}
7218 \providetranslation{Notation (glossaries)}{Nota\c c\~ao}
7219 \providetranslation{Description (glossaries)}{Descri\c c\~ao}
7220 \providetranslation{Symbol (glossaries)}{S\`imbolo}
7221 \providetranslation{Page List (glossaries)}{Lista de P\`aginas}
7222 \providetranslation{Symbols (glossaries)}{S\`imbolos}
7223 \providetranslation{Numbers (glossaries)}{N\`umeros}
```

### 6.4 Danish Dictionary

This is a dictionary file provided for use with the package.

```
7224 \ProvidesDictionary{glossaries-dictionary}{Danish}
```

Provide Danish translations:

```
7225 \providetranslation{Glossary}{Ordliste}
7226 \providetranslation{Acronyms}{Akronymer}
7227 \providetranslation{Notation (glossaries)}{Symbolforklaring}
7228 \providetranslation{Description (glossaries)}{Beskrivelse}
7229 \providetranslation{Symbol (glossaries)}{Symbol}
7230 \providetranslation{Page List (glossaries)}{Side}
7231 \providetranslation{Symbols (glossaries)}{Symboler}
7232 \providetranslation{Numbers (glossaries)}{Tal}
```

### 6.5 Dutch Dictionary

This is a dictionary file provided for use with the package.

```
7233 \ProvidesDictionary{glossaries-dictionary}{Dutch}
```

Provide Dutch translations:

```
7234 \providetranslation{Glossary}{Woordenlijst}
7235 \providetranslation{Acronyms}{Acroniemen}
7236 \providetranslation{Notation (glossaries)}{Benaming}
7237 \providetranslation{Description (glossaries)}{Beschrijving}
7238 \providetranslation{Symbol (glossaries)}{Symbool}
7239 \providetranslation{Page List (glossaries)}{Pagina's}
7240 \providetranslation{Symbols (glossaries)}{Symbolen}
7241 \providetranslation{Numbers (glossaries)}{Cijfers}
```

### 6.6 English Dictionary

This is a dictionary file provided for use with the package.

```
7242 \ProvidesDictionary{glossaries-dictionary}{English}
```

Provide English translations:

```
7243 \providetranslation{Glossary}{Glossary}
7244 \providetranslation{Acronyms}{Acronyms}
7245 \providetranslation{Notation (glossaries)}{Notation}
7246 \providetranslation{Description (glossaries)}{Description}
7247 \providetranslation{Symbol (glossaries)}{Symbol}
7248 \providetranslation{Page List (glossaries)}{Page List}
7249 \providetranslation{Symbols (glossaries)}{Symbols}
7250 \providetranslation{Numbers (glossaries)}{Numbers}
```

## 6.7 French Dictionary

This is a dictionary file provided for use with the package.

```
7251 \ProvidesDictionary{glossaries-dictionary}{French}
```

Provide French translations:

```
7252 \providetranslation{Glossary}{Glossaire}
7253 \providetranslation{Acronyms}{Acronymes}
7254 \providetranslation{Notation (glossaries)}{Terme}
7255 \providetranslation{Description (glossaries)}{Description}
7256 \providetranslation{Symbol (glossaries)}{Symbole}
7257 \providetranslation{Page List (glossaries)}{Pages}
7258 \providetranslation{Symbols (glossaries)}{Symboles}
7259 \providetranslation{Numbers (glossaries)}{Nombres}
```

## 6.8 German Dictionary

This is a dictionary file provided for use with the package.

```
7260 \ProvidesDictionary{glossaries-dictionary}{German}
```

Provide German translations (quite a few variations were suggested for German; I settled on the following):

```
7261 \providetranslation{Glossary}{Glossar}
7262 \providetranslation{Acronyms}{Akronyme}
7263 \providetranslation{Notation (glossaries)}{Bezeichnung}
7264 \providetranslation{Description (glossaries)}{Beschreibung}
7265 \providetranslation{Symbol (glossaries)}{Symbol}
7266 \providetranslation{Page List (glossaries)}{Seiten}
7267 \providetranslation{Symbols (glossaries)}{Symbole}
7268 \providetranslation{Numbers (glossaries)}{Zahlen}
```

## 6.9 Irish Dictionary

This is a dictionary file provided for use with the package.

```
7269 \ProvidesDictionary{glossaries-dictionary}{Irish}
```

Provide Irish translations:

```
7270 \providetranslation{Glossary}{Gluais}
7271 \providetranslation{Acronyms}{Acrainmneacha}
```

```

7272 \providetranslation{Notation (glossaries)}{Ciall}
7273 \providetranslation{Description (glossaries)}{Tuairisc}
7274 \providetranslation{Symbol (glossaries)}{Comhartha}
7275 \providetranslation{Page List (glossaries)}{Leathanaigh}
7276 \providetranslation{Symbols (glossaries)}{Comhartha'\{i}}
7277 \providetranslation{Numbers (glossaries)}{Uimhreacha}

```

## 6.10 Italian Dictionary

This is a dictionary file provided for use with the package.

```
7278 \ProvidesDictionary{glossaries-dictionary}{Italian}
```

Provide Italian translations:

```

7279 \providetranslation{Glossary}{Glossario}
7280 \providetranslation{Acronyms}{Acronimi}
7281 \providetranslation{Notation (glossaries)}{Nomenclatura}
7282 \providetranslation{Description (glossaries)}{Descrizione}
7283 \providetranslation{Symbol (glossaries)}{Simbolo}
7284 \providetranslation{Page List (glossaries)}{Elenco delle pagine}
7285 \providetranslation{Symbols (glossaries)}{Simboli}
7286 \providetranslation{Numbers (glossaries)}{Numeri}

```

## 6.11 Magyar Dictionary

This is a dictionary file provided for use with the package.

```
7287 \ProvidesDictionary{glossaries-dictionary}{Magyar}
```

Provide translations:

```

7288 \providetranslation{Glossary}{Sz\'ojegyz\'ek}
7289 \providetranslation{Acronyms}{Bet\'H uszavak}
7290 \providetranslation{Notation (glossaries)}{Kifejez\'es}
7291 \providetranslation{Description (glossaries)}{Magyar\'azat}
7292 \providetranslation{Symbol (glossaries)}{Jel\'ol\'es}
7293 \providetranslation{Page List (glossaries)}{Oldalsz\'am}
7294 \providetranslation{Symbols (glossaries)}{Jelek}
7295 \providetranslation{Numbers (glossaries)}{Sz\'amjegyek}

```

## 6.12 Polish Dictionary

This is a dictionary file provided for use with the package.

```
7296 \ProvidesDictionary{glossaries-dictionary}{Polish}
```

Provide Polish translations:

```

7297 \providetranslation{Glossary}{S\lownik termin\'ow}
7298 \providetranslation{Acronyms}{Skr\'ot}
7299 \providetranslation{Notation (glossaries)}{Termin}
7300 \providetranslation{Description (glossaries)}{Opis}
7301 \providetranslation{Symbol (glossaries)}{Symbol}
7302 \providetranslation{Page List (glossaries)}{Strony}

```

```
7303 \providetranslation{Symbols (glossaries)}{Symbole}
7304 \providetranslation{Numbers (glossaries)}{Liczby}
```

## 6.13 Serbian Dictionary

This dictionary was provided by Zoran Filipovic.

```
7305 \ProvidesDictionary{glossaries-dictionary}{Serbian}
7306 \providetranslation{Glossary}{Mali re\vnik}
7307 \providetranslation{Acronyms}{Skra\'cenice}
7308 \providetranslation{Notation (glossaries)}{Oznaka}
7309 \providetranslation{Description (glossaries)}{Opis}
7310 \providetranslation{Symbol (glossaries)}{Simbol}
7311 \providetranslation{Page List (glossaries)}{Stranica}
7312 \providetranslation{Symbols (glossaries)}{Simboli}
7313 \providetranslation{Numbers (glossaries)}{Brojevi}
```

## 6.14 Spanish Dictionary

This is a dictionary file provided for use with the package.

```
7314 \ProvidesDictionary{glossaries-dictionary}{Spanish}
```

Provide Spanish translations:

```
7315 \providetranslation{Glossary}{Glosario}
7316 \providetranslation{Acronyms}{Siglas}
7317 \providetranslation{Notation (glossaries)}{Entrada}
7318 \providetranslation{Description (glossaries)}{Descripci\'on}
7319 \providetranslation{Symbol (glossaries)}{S\'mbolo}
7320 \providetranslation{Page List (glossaries)}{Lista de p\'aginas}
7321 \providetranslation{Symbols (glossaries)}{S\'mbolos}
7322 \providetranslation{Numbers (glossaries)}{N\'umeros}
```

## Glossary

`makeindex` An indexing application. 17

`xindy` An flexible indexing application with multilingual support written in Perl. 17

## Change History

1.01		'delimN and 'delimR in ist file	118
General: Added range facility in	1.03		
format key .....	61	\makefirstuc: changed 'pro-	
\writeist: Added spaces after		tected@edef to 'def .....	170

1.04	General: Added 'glstextformat ... 56	numberedsection: numbered-section package option added . 6
1.05	\glossarysection: added '@mk-both to 'glossarysection .... 25	numberline: numberline option added ..... 5
	\glsmakefirsttuc: new ..... 171	1.12
	\newglossaryentry: Changed the default value of the sort key to just the value of the name key ..... 48	\@GLSpl: now uses 'glsentrydescplural and 'glsentrysymbolplural instead of 'glsentrydesc and 'glsentrysymbol ..... 77
1.06	General: now requires etoolbox . 170	\@GLspl@: now uses 'glsentrydescplural and 'glsentrysymbolplural instead of 'glsentrydesc and 'glsentrysymbol ..... 76
	\capitalisewords: new ..... 171	\@glspl@: now uses 'glsentrydescplural and 'glsentrysymbolplural instead of 'glsentrydesc and 'glsentrysymbol ..... 74
	\xcapitalisewords: new .... 172	General: added check for 'hypertarget separate to 'hyperlink (memoir defines 'hyperlink but not 'hypertarget) ... 69
1.07	\@gls@link: fixed bug caused by \theglsentrycounter setting the page number too soon 59	fixed bug ('GLSdesc shouldn't use 'gls@<type>@display) .... 88
	\glsadd: fixed bug caused by \theglsentrycounter setting the page number too soon ..... 116	fixed bug ('Glsdesc shouldn't use 'gls@<type>@display) .... 88
1.08	General: Added babel support ... 21	fixed bug ('GLSdesc shouldn't use 'gls@<type>@display) ..... 87
	listgroup: changed listgroup style to use \glsgetgrouptitle ..... 177	fixed bug ('GLSfirst shouldn't use 'gls@<type>@display) .... 82
	altlistgroup: changed altlistgroup style to use \glsgetgrouptitle ..... 178	fixed bug ('Glsfirst shouldn't use 'gls@<type>@display) ..... 82
	\newglossaryentry: Fixed error message to say "description key" rather than "desc key" .. 47	fixed bug ('glsfirst shouldn't use 'gls@<type>@display) ..... 81
1.1	\@glossarysection: numbered sections and auto label added 27	fixed bug ('GLSfirstplural shouldn't use 'gls@<type>@display) ..... 85
	\@gls@tmpb: changed \toksdef to \newtoks ..... 63	fixed bug ('Glsfirstplural shouldn't use 'gls@<type>@display) ..... 85
	\@gls@toc: numberline added .. 28	fixed bug ('glsfirstplural shouldn't use 'gls@<type>@display) ..... 84
	\@p@glossarysection: numbered sections and auto label added ..... 27	fixed bug ('GLSname shouldn't use 'gls@<type>@display) .... 87
	General: Added support for translator package ..... 21	fixed bug ('glsname shouldn't use 'gls@<type>@display) .... 86
	amsgen now loaded (\new@ifnextchar needed) ..... 3	fixed bug ('GLSplural shouldn't use 'gls@<type>@display) .... 84
	translate: translate option added ..... 15	
	\setglossarysection: new ... 27	



fixed bug ('Glsplural shouldn't use 'gls@<type>@display) ....	83	'newacronym in the preamble	147
fixed bug ('glsplural shouldn't use 'gls@<type>@display) ....	83	\newglossaryentry: Changed default first value .....	48
fixed bug ('GLSymbol shouldn't use 'gls@<type>@display) .....	91	Changed default firstplural value .....	48
fixed bug ('Glssymbol shouldn't use 'gls@<type>@display) ....	91	Removed restriction on only using 'newglossaryentry in the preamble .....	52
fixed bug ('glssymbol shouldn't use 'gls@<type>@display) ....	90	1.14	
fixed bug ('glssymbolplural shouldn't use 'gls@<type>@display) .....	92	\@gls@hypergroup: new .....	173
fixed bug ('GLStext shouldn't use 'gls@<type>@display) ....	80	General: added nonnumberlist key to 'printglossary .....	136
fixed bug ('Glstext shouldn't use 'gls@<type>@display) .....	81	added numberedsection key to 'printglossary .....	136
fixed bug ('glstext shouldn't use 'gls@<type>@display) .....	80	\firstacronymfont: new ....	150
descriptionplural: new .....	43	\glsautoprefix: new .....	6
\Glsentrydescplural: New ..	110	\glsnavhyperlink: changed 'edef to 'protected@edef ...	172
\Glsentrydescplural: New ..	110	\glsnavhypertarget: added write to aux file .....	173
\Glsentrysymbolplural: New	111	\glsnavigation: changed to only use labels for groups that are present .....	174
\Glsentrysymbolplural: New	111	1.15	
\newglossaryentry: Changed default first plural to be first key with s appended (was text key with s appended) .....	48	\@gls@link: added 'glslabel ....	59
descriptionplural support added .....	48	General: Added 'glssettoctitle ...	21
symbolplural support added ..	48	\gls@hypergroup: new ..	173
\SetDescriptionFootnoteAcronymStyle: Added 'protect before 'footnote and 'glslink .....	153	\glsnavhypertarget: added check if rerun required ....	173
\SetFootnoteAcronymStyle: Added 'protect before 'footnote and 'glslink .....	157	\glssettoctitle: new .....	20
symbolplural: new .....	44	\newglossaryentry: check for '@glo@first in description ...	51
1.13		check for '@glo@text in symbol	51
General: Add Polish support	248, 251	\printglossary: changed the way the TOC title is set ....	133
fixed bug that ignores 3rd parameter .....	80–93	1.16	
\ACRfullpl: new .....	149	\@GLS@: Test glossary type is 'acronymtype in addition to checking if footnote option has been used .....	73, 231
\Acrfullpl: new .....	149	\@GLSpl: Test glossary type is 'acronymtype in addition to checking if footnote option has been used .....	78
\acrfullpl: new .....	149	\@Gls@: Test glossary type is 'acronymtype in addition to checking if footnote option has been used .....	72
\acrpluralsuffix: New .....	147		
\newacronym: Removed restriction on only using			

\@Glspl@: Test glossary type is 'acronym' type in addition to checking if footnote option has been used .....	76	\glshypernumber: modified to allow material to be attached to location .....	143
\@gls@: Test glossary type is 'acronym' type in addition to checking if footnote option has been used .....	71	\glsnavhyperlink: replaced 'hy- perlink to '@glslink .....	172
\@gls@pl@: Test glossary type is 'acronym' type in addition to checking if footnote option has been used .....	232	\glsnavhypertarget: replaced 'hypertarget to '@glstarget ..	173
\@glsdisp: Test glossary type is 'acronym' type in addition to checking if footnote option has been used .....	79	\glssee: new .....	131
\@glspl@: Test glossary type is 'acronym' type in addition to checking if footnote option has been used .....	75	\glsseeformat: new .....	131
\@glstarget: raised the hyper- target so the target text doesn't scroll off the top of the page ..	69	\glsSetSuffixF: new .....	24
\newglossaryentry: Changed def to let .....	48	\glsSetSuffixFF: new .....	24
Changed if to ifx .....	50	\ifglxindy: new .....	17
1.17		\istfilename: added xindy sup- port .....	23
\@do@seeglossary: new .....	130	\newglossaryentry: added non- umberlist key .....	48
\@glo@storeentry: new .....	53	added parent key .....	48
\@glossary: changed defini- tion to use \index instead of \@index .....	127	added see key .....	48
\@glsdefaultplural: new ....	47	Stored main part of entry format when entry is defined .....	52
\@glsdefaultsort: new .....	47	\newglossarystyle: made 'new- glossarystyle long .....	142
\@glshypernumber: new .....	143	\nopostdesc: new .....	22
\@glsnoname: new .....	46	nonumberlist: new .....	45
\@glsnonextpages: new .....	137	\printglossary: added check to determine if 'printglossary is already defined .....	133
\@wrglossary: modified to allow for xindy support .....	127	added print language to aux file	133
General: added Brazilian dictio- nary .....	252	order: order package option added .....	16
Added Brazilian support ....	248	\writeist: added xindy support	118
added xindy support .....	16	1.18	
parent: new .....	45	\@gls@loadlist: new .....	7
see: new .....	45	\@gls@loadlong: new .....	7
\gls@romanpage: new .....	128	\@gls@loadsuper: new .....	7
\gls@suffixF: new .....	24	\@gls@loadtree: new .....	8
\gls@suffixFF: new .....	24	\glstarget: new .....	140
\glshyperlink: new .....	115	\newglossaryentry: Changed default value of sort to '@gls- defaultsort .....	48
		moved sort sanitization to 'new- glossaryentry .....	51
		\oldacronym: new .....	146
		nolist: new .....	7
		nolong: new .....	7
		sort: moved sanitization to 'new- glossaryentry .....	43
		nostyles: new .....	8
		nosuper: new .....	7

notree: new .....	8	2.03	
1.19			\@GLS@: Added check for hyper-
\glsclearpage: new .....	28		first .....
\glsdisp: new .....	78		\@GLSpl: Added check for hyper-
\SetDescriptionAcronymStyle:			first .....
changed 'acronymfont to use			\@Gls@: Added check for hyper-
'textsmaller instead of 'smaller	156		first .....
\SetDescriptionFootnoteAcronymStyle\@Glspl@:			Added check for hyper-
changed 'acronymfont to use			first .....
'textsmaller instead of 'smaller	153		\@Gls@: Added check for hyper-
\SetFootnoteAcronymStyle:			first .....
changed 'acronymfont to use			\@Gls@@link: new .....
'textsmaller instead of 'smaller	157		\@Gls@link: added \leavevmode
\SetSmallAcronymStyle:			.....
changed 'acronymfont to use			Moved entry existence check to
'textsmaller instead of 'smaller	159		avoid .....
1.2			\@glsdisp: Added check for hy-
General: fixed bug in ngerman			perfirst .....
captions .....	245		\@glspl@: Added check for hyper-
2.01			first .....
\@Gls@link: moved \@do@wrglossary			\@glossarymark: Added check to
before term is displayed to pre-			see if it's already defined ....
vent unwanted whatsit .....	60		hyperfirst: new .....
General: added nomain package		2.04	
option .....	11		\@GLS@: Changed test to check if
\forall glossaries: replaced			glossary type has been identi-
\ifthenelse with \ifx ....	38		fied as a list of acronyms ....
\forgl sentries: replaced			\@GLSpl: Changed test to check if
\ifthenelse with \ifx ....	38		glossary type has been identi-
\glstdefmain: new .....	11		fied as a list of acronyms ....
\glstdescwidth: changed			\@Gls@: Changed test to check if
\linewidth to \hsize . 180, 194			glossary type has been identi-
\glstlistdottedwidth: changed			fied as a list of acronyms ....
\linewidth to \hsize ....	179		\@GLspl@: Changed test to check
\glstpagelistwidth: changed			if glossary type has been iden-
\linewidth to \hsize . 180, 195			tified as a list of acronyms ...
\writeist: removed item_02 - no			\@glossaryentryfield: new ..
such makeindex key .....	122		\@glossarysubentryfield:
2.02			new .....
General: Changed Brazil to Brazil-			\@Gls@: Changed test to check if
ian .....	252		glossary type has been identi-
false will prevent automatic			fied as a list of acronyms ....
loading of translator package	18		\@Glsacronymlists: new .....
\glossarymark: New .....	26		\@glstisp: Changed test to check
\glossarysection: changed			if glossary type has been iden-
'@mkboth to 'glossarymark ..	25		tified as a list of acronyms ...
\printglossary: suppressed			\@glstpl@: Changed test to check
warning globally rather than			if glossary type has been iden-
locally .....	135		tified as a list of acronyms ...

\@newglossaryentryposthook: new ..... 52	\writeist: Added \string before opening and closing braces. Patch provided by Segiu Dotenco ..... 123
\@newglossaryentryprehook: new ..... 52	
acronymlists:new ..... 13	2.06
\DeclareAcronymList:new ... 12	\altnewglossary:new ..... 42
\DefineAcronymSynonyms:new 163	\CustomAcronymFields:new . 161
\glsadd: fixed bug that ignored counter ..... 116	\CustomNewAcronymDef:new . 162
\Glsentryuseri:new ..... 112	\SetCustomDisplayStyle:new 161
\Glsentryuseri:new ..... 112	\SetCustomStyle:new ..... 162
\Glsentryuserii:new ..... 112	2.07
\Glsentryuserii:new ..... 112	General: glssadd format key stored in \@glsnumberformat (was mistakenly stored in \@glo@format) ..... 116
\Glsentryuseriii:new ..... 112	
\Glsentryuseriii:new ..... 112	3.0
\Glsentryuseriv:new ..... 112	\@@glossarysec: replaced with \@ifundefined ..... 5
\Glsentryuseriv:new ..... 112	\@do@seeglossary: Sanitize and escape cross-referencing information ..... 130
\Glsentryuservi:new ..... 113	\@gls@counterwithin:new .... 9
\Glsentryuservi:new ..... 113	\@gls@ifinlist:new ..... 29
\newglossary: added check to determine if \@gls@<type>@display and \@gls@<type>@displayfirst have been defined. .... 41	\@gls@link: added \@gls@saveentrycounter ..... 59
\newglossaryentry: added user1-6 keys ..... 48	added \@gls@setsort ..... 60
\SetAcronymLists:new ..... 13	\@gls@saveentrycounter:new 60
\SetDefaultAcronymDisplayStyle:new ..... 150	\@gls@setupsort@def:new ... 10
\SetDefaultAcronymStyle:new ..... 151	\@gls@setupsort@standard:new ..... 9
\SetDescriptionAcronymDisplayStyle:new ..... 155	\@gls@setupsort@use:new ... 10
\SetDescriptionDUAAcronymDisplayStyle:new ..... 154	\@gls@xdy@locationlist:new 32
\SetDescriptionFootnoteAcronymDisplayStyle:new ..... 152	\@gls@link: replaced \@ifundefined with \ifcsundef ..... 69
\SetDUADisplayStyle:new .. 160	\@gls@nextpages:new ..... 137
\SetFootnoteAcronymDisplayStyle:new ..... 156	\@set@gl: Added check for savewrites ..... 124
\SetSmallAcronymDisplayStyle:new ..... 158	\@set@glo@numformat: added 4th argument ..... 61
2.05	\@wrglossary: modified to take into account savewrites .... 127
\@glsdisp: Added closing brace. Patch provided by Sergiu Dotenco ..... 79	\@xdy@attributelist:new .... 28
Removed spurious brace. Patch provided by Sergiu Dotenco . 79	General: added prefix to hyperlink ..... 145
	etoolbox now loaded ..... 4
	replaced \@ifundefined with \ifcsundef ..... 19, 58, 136
	\acrfootnote:new ..... 152

<code>\ACRfull:</code> added starred version	148	<code>\glsgetgrouptitle:</code>	re-
<code>\Acrfull:</code> added starred version	148	placed <code>\@ifundefined</code> with	
<code>\acrfull:</code> added starred version	147	<code>\ifcsundef</code>	141
<code>\ACRfullpl:</code> added starred ver-		<code>\glshyperlink:</code> changed de-	
sion	150	fault from <code>\glstryname</code> to	
<code>\Acrfullpl:</code> added starred ver-		<code>\glstrytext</code>	115
sion	149	<code>\glshypernumber:</code> replaced	
<code>\acrfullpl:</code> added starred ver-		<code>\@ifundefined</code> with	
sion	149	<code>\ifcsundef</code>	143
<code>\acrlinkfootnote:</code> new	152	<code>\glsnumberformat:</code> replaced	
<code>\acrnolinkfootnote:</code> new	152	<code>\@ifundefined</code> with	
<code>\addglossarytocaptions:</code> re-		<code>\ifcsundef</code>	24
placed <code>\@ifundefined</code> with		<code>\glsrefentry:</code> new	138
<code>\ifcsundef</code>	21	<code>\glsresetsubentrycounter:</code>	
<code>savewrites:</code> new	17	new	138
<code>see:</code> added <code>\@glo@seeautonumberlist</code>		<code>\glsseeitem:</code> hyperlink uses	
	45	<code>\glsseeitemformat</code> instead	
<code>seeautonumberlist:</code> new	7	of <code>\glstryname</code>	132
<code>\glossarymark:</code> replaced		<code>\glsseeitemformat:</code> new	132
<code>\@ifundefined</code> with		<code>\glssortnumberfmt:</code> new	9
<code>\ifcsundef</code>	26	<code>\glsstepentry:</code> new	138
<code>\glossarysection:</code> replaced		<code>\glsstepsubentry:</code> new	138
<code>\@ifundefined</code> with		<code>\glssubentrycounterlabel:</code>	
<code>\ifcsundef</code>	25	new	139
<code>\glossarystyle:</code> replaced		<code>\glssubentryitem:</code> new	139
<code>\@ifundefined</code> with		<code>theglossary:</code> replaced <code>\@ifundefined</code>	
<code>\ifcsundef</code>	142	with <code>\ifcsundef</code>	139
<code>\gls@codepage:</code> replaced		<code>short:</code> new	46
<code>\@ifundefined</code> with		<code>shortplural:</code> new	46
<code>\ifcsundef</code>	17	<code>\ifglossaryexists:</code> re-	
<code>\gls@doclearpage:</code> replaced		placed <code>\@ifundefined</code> with	
<code>\@ifundefined</code> with		<code>\ifcsundef</code>	39
<code>\ifcsundef</code>	27	<code>\ifglstryexists:</code> re-	
<code>\gls@romanpage:</code> added check		placed <code>\@ifundefined</code> with	
for hyper location prefix	129	<code>\ifcsundef</code>	39
modified to use new format	128	<code>\istfile:</code> deprecated	126
<code>\glsadd:</code> added <code>\@gls@saveentrycounter</code>		<code>glossaryentry:</code> new	137
	116	<code>glossarysubentry:</code> new	137
<code>\GlsAddXdyCounters:</code> new	29	<code>\newglossary:</code> added <code>\@gls@defsortcount</code>	
<code>\glstrycounterlabel:</code> new	139		42
<code>\glstryitem:</code> new	139	replaced <code>\@ifundefined</code> with	
<code>\Glsentrylong:</code> new	113	<code>\ifcsundef</code>	41
<code>\glstrylong:</code> new	113	<code>\newglossaryentry:</code> added	
<code>\Glsentrylongpl:</code> new	114	<code>\@gls@defsort</code>	51
<code>\glstrylongpl:</code> new	113	added short and long keys	48
<code>\Glsentryshort:</code> new	113	replaced <code>\@ifundefined</code> with	
<code>\glstryshort:</code> new	113	<code>\ifcsundef</code>	48
<code>\Glsentryshortpl:</code> new	113	replaced <code>\DeclareRobustCommand</code>	
<code>\glstryshortpl:</code> new	113	with <code>\newrobustcmd</code>	47

<code>\newglossarystyle:</code>	re-	<code>\showglouserii:</code>	new	166
placed <code>\@ifundefined</code> with		<code>\showglouseriii:</code>	new	166
<code>\ifcsundef</code>	142	<code>\showglouseriv:</code>	new	167
<code>entrycounter:</code>	new	<code>\showglouserv:</code>	new	167
<code>entrycounterwithin:</code>	new	<code>\showglouservi:</code>	new	167
<code>\oldacronym:</code>	replaced <code>\@ifundefined</code>	<code>subentrycounter:</code>	new	9
with <code>\ifcsundef</code>	146	<code>\writeist:</code>	added	xindy-only
<code>compatible-2.07:</code>	compatible-	macro definitions to glossary		
<code>2.07 option added</code>	18	open tag	120	
<code>long:</code>	new	modified to support new for-		
<code>longplural:</code>	new	mat	118	
<code>nonumberlist:</code>	now boolean			
<code>sort:</code>	new			
<code>counter:</code>	replaced <code>\@ifundefined</code>	General:	made robust	73
with <code>\ifcsundef</code>	45	<code>\ACRfull:</code>	made robust	148
<code>\printglossary:</code>	added	<code>\Acrfull:</code>	made robust	148
<code>\currentglossary</code>	134	<code>\acrfull:</code>	made robust	147
added <code>\glsnexpages</code>	134	<code>\acrfullformat:</code>	removed	
make to title default to title	133	<code>\acronymfont:</code>	as it should al-	
replaced <code>\@ifundefined</code> with		ready be set in the second ar-		
<code>\ifcsundef</code>	133, 134	gument.	148	
<code>\SetDescriptionFootnoteAcronymDisplayStyle</code>		<code>\ACRfullpl:</code>	made robust	149
expanded options link op-		<code>\Acrfullpl:</code>	made robust	149
tions	152	<code>\acrfullpl:</code>	made robust	149
<code>\setentrycounter:</code>	added op-	<code>\ACRlong:</code>	made robust	107
tional argument	142	<code>\Acrlong:</code>	made robust	106
<code>\showacronymlists:</code>	new	<code>\acrlong:</code>	made robust	106
<code>\showglocounter:</code>	new	<code>\ACRlongpl:</code>	made robust	109
<code>\showglodesc:</code>	new	<code>\Acrlongpl:</code>	made robust	108
<code>\showglodescplural:</code>	new	<code>\acrlongpl:</code>	made robust	107
<code>\showglofirst:</code>	new	<code>\ACRshort:</code>	made robust	103
<code>\showglofirstpl:</code>	new	<code>\Acrshort:</code>	made robust	103
<code>\showgloflag:</code>	new	<code>\acrshort:</code>	made robust	102
<code>\showgloindex:</code>	new	<code>\ACRshortpl:</code>	made robust	105
<code>\showglolevel:</code>	new	<code>\Acrshortpl:</code>	made robust	104
<code>\showgloname:</code>	new	<code>\acrshortpl:</code>	made robust	104
<code>\showgloparent:</code>	new	<code>\Gls:</code>	made robust	71
<code>\showgloplural:</code>	new	<code>\glsadd:</code>	made robust	116
<code>\showglosort:</code>	new	<code>\glsaddall:</code>	made robust	116
<code>\showglossaries:</code>	new	<code>\GLSdesc:</code>	made robust	88
<code>\showglossarycounter:</code>	new	<code>\Glsdesc:</code>	made robust	88
<code>\showglossaryentries:</code>	new	<code>\glsdesc:</code>	made robust	87
<code>\showglossaryin:</code>	new	<code>\GLSdescplural:</code>	made robust	90
<code>\showglossaryout:</code>	new	<code>\Glsdescplural:</code>	made robust	89
<code>\showglossarytitle:</code>	new	<code>\glsdescplural:</code>	made robust	89
<code>\showglosymbol:</code>	new	<code>\glsfirst:</code>	made robust	81
<code>\showglosymbolplural:</code>	new	<code>\GLSfirstplural:</code>	made robust	85
<code>\showglotext:</code>	new	<code>\Glsfirstplural:</code>	made robust	85
<code>\showglotype:</code>	new	<code>\glsfirstplural:</code>	made robust	84

\glslink: made robust .....	59	savenumberlist: new .....	7
\GLSname: made robust .....	87	ucmark: new .....	8
\Glsname: made robust .....	86	\gls@romanpage: changed	
\glsname: made robust .....	86	\@glslocref to \theglsentrycounter	
\GLSpl: made robust .....	77	.....	130
\Glspl: made robust .....	75	\gls@save@numberlist: new ..	132
\glspl: made robust .....	74	\glsdisplaynumberlist: new	114
\GLSplural: made robust .....	84	\glsentrycounter: set default	
\GLSsymbol: made robust .....	91	value .....	60
\GLssymbol: made robust .....	91	\Glsentryfull: fixed bug (re-	
\glsymbol: made robust .....	90	placed \glsentryshortpl	
\GLSsymbolplural: made robust	93	with \glsentryshort) ....	114
\GLssymbolplural: made robust	92	\glsentryfullpl: fixed bug (re-	
\glsymbolplural: made robust	92	placed \glsentryshort with	
\Gls text: made robust .....	81	\glsentryshortpl) .....	114
\gls text: made robust .....	80	\glsentrynumberlist: new ..	114
\GLSuseri: made robust .....	94	\glsmoveentry: new .....	52
\Glsuseri: made robust .....	94	\glsnumlistlastsep: new ...	115
\glsuseri: made robust .....	93	\glsnumlistsep: new .....	115
\GLSuserii: made robust .....	96	\glsresetsubentrycounter:	
\Glsuserii: made robust .....	95	new .....	138
\glsuserii: made robust .....	95	\glswritefiles: added check	
\GLSuseriii: made robust .....	97	for existence of token in case	
\Glsuseriii: made robust .....	97	\makeglossaries has been	
\glsuseriii: made robust .....	96	omitted .....	126
\GLSuseriv: made robust .....	99	\ifglshaschildren: new .....	40
\Glsuseriv: made robust .....	98	\ifglshasparent: new .....	40
\glsuseriv: made robust .....	98	\makeglossaries: added list	
\GLSuserv: made robust .....	100	parser .....	125
\Glsuserv: made robust .....	100	indexonlyfirst: new .....	15
\glsuserv: made robust .....	99	\newglossaryentry: added	
\GLSuservi: made robust ....	102	numberlist element .....	51
\Glsuservi: made robust ....	101	\printglossary: add a way to	
\glsuservi: made robust ....	101	fetch current entry label ...	134
\glswritefiles: added check		\renewglossarystyle: new ..	142
for empty glossaries .....	126	\showglossaryentries: fixed	
3.02		misspelt command .....	169
\@do@wrglossary: changed		\SmallNewAcronymDef: fixed	
\@do@wr@glossary to test for		broken short and long plural	158
indexonlyfirst option; put old		3.03	
\@do@wr@glossary code into		\@gls@sanitizesort: new ....	14
\@do@wrglossary .....	128	\@gls@setupsort@standard:	
\@gls@missingnumberlist:		used \@gls@sanitizesort ..	9
new .....	47	General: allow title to set toctitle	136
\@wrglossary: added check for		altlongragged4col: added	
glossary file defined .....	127	check for glsnogroupskip ..	189
General: added check for polyglos-		altsuperragged4col: added	
sia .....	18	check for glsnogroupskip ..	205
reversed order of package check	22		

alttree: added check for	tize	\gls@numberpage,
glsnogroupskip .....	\gls@alphpage, \gls@Alphpage	
index: added check for	and \gls@romanpage .....	62
glsnogroupskip .....	General: Added check for doc	
nogroupskip: new .....	package .....	4
8	added datatool-base as a re-	
long: added check for	quired package .....	3
glsnogroupskip .....	added local key .....	58
180	\changes: new .....	4
long3col: added check for	\gls@Alphpage: new .....	128
glsnogroupskip .....	\gls@alphpage: new .....	128
182	\gls@disablepagerefexpansion:	
long4col: added check for	new .....	128
glsnogroupskip .....	\gls@numberpage: new .....	128
183	\gls@protected@pagefmts:	
longragged: added check for	new .....	128
glsnogroupskip .....	\gls@romanpage: changed	
186	\theglsentrycounter back	
longragged3col: added check	to \@glslocref .....	130
for glsnogroupskip .....	new .....	128
188	\glsdefmain: added check for	
nopostdot: new .....	doc package .....	11
8	\glsorg@endtheglossary: new .	5
\printglossary: allow title to	\glsorg@glossary: new .....	4
override default toctitle ....	\glsorg@theglossary: new ....	4
133	\glsorg@wrglossary: new .....	4
tree: added check for	altlist: replaced \newline with	
glsnogroupskip .....	paragraph break .....	178
209	\PrintChanges: new .....	5
treenoname: added check for	\printglossary: Moved aux	
glsnogroupskip .....	write to end of document to	
210	prevent unwanted whatsit oc-	
super: added check for	curing here. ....	134
glsnogroupskip .....		
195		
super3col: added check for		
glsnogroupskip .....		
197		
super4col: added check for		
glsnogroupskip .....		
198		
superragged: added check for		
glsnogroupskip .....		
202		
superragged3col: added check		
for glsnogroupskip .....		
203		
3.04		
\@gls@escbsdq: unsani-		



## 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.

<b>Symbols</b>	
\@glossarysec .....	5
\@glossaryseclabel .....	6
\@glossarysecstar .....	5
\@ACRlong .....	236
\@ACRshort .....	235
\@Acrlong .....	236
\@Acrshort .....	235
\@GLS@ .....	73, 231
\@GLSpl .....	77
\@GLSpl@ .....	234
\@Gls@ .....	71, 230
\@Glspl@ .....	76, 233
\@acrlong .....	236
\@acrshort .....	235
\@addtoacronymlists .....	12
\@delimN .....	144
\@delimR .....	144
\@disable@onlypremakeg .....	19
\@disable@premakecs .....	19
\@disabled@gl saddxdycounters ..	29
\@do@seeglossary .....	130
\@do@wrglossary .....	128, 215
\@glo@seeautonumberlist .....	7
\@glo@storeentry .....	53
\@glo@types .....	40
\@glossary .....	127
\@glossary@default@style .....	6
\@glossaryentryfield .....	53, 237
\@glossarysection .....	27
\@glossarysubentryfield ..	53, 237
\@gls .....	70
\@gls@ .....	70, 229
\@gls@@link .....	59
\@gls@addpredefinedattributes	31
\@gls@checkactual .....	67
\@gls@checkbar .....	66
\@gls@checkescactual .....	64
\@gls@checkescbar .....	65
\@gls@checkesclevel .....	65
\@gls@checkescquote .....	64
\@gls@checklevel .....	66
\@gls@checkmkidxchars .....	62
\@gls@checkquote .....	63
\@gls@codepage .....	37
\@gls@counterwithin .....	9
\@gls@escbsdq .....	62
\@gls@fixbraces .....	131
\@gls@getcounter .....	42
\@gls@getcounterprefix .....	130
\@gls@hypergroup .....	173
\@gls@ifinlist .....	29
\@gls@link .....	59
\@gls@loadlist .....	7
\@gls@loadlong .....	7
\@gls@loadsuper .....	7
\@gls@loadtree .....	8
\@gls@missingnumberlist .....	47
\@gls@noaccess .....	223
\@gls@onlypremakeg .....	19
\@gls@pl@ .....	232
\@gls@renewglossary .....	127
\@gls@sanitizedesc .....	14
\@gls@sanitizename .....	14
\@gls@sanitizesort .....	14
\@gls@sanitizesymbol .....	14
\@gls@saveentrycounter .....	60
\@gls@setcounter .....	42
\@gls@setupsort@def .....	10
\@gls@setupsort@standard .....	9
\@gls@setupsort@use .....	10
\@gls@tmpb .....	63
\@gls@toc .....	28
\@gls@updatechecked .....	63
\@gls@xdy@Lclass@Alpha-page-numbers	33
\@gls@xdy@Lclass@Appendix-page-numbers	33
\@gls@xdy@Lclass@Roman-page-numbers	32
\@gls@xdy@Lclass@alpha-page-numbers	32
\@gls@xdy@Lclass@arabic-page-numbers	32
\@gls@xdy@Lclass@arabic-section-numbers	33



<code>\addglossarytocaptions</code> .....	21	<code>\DefineAcronymSynonyms</code> .....	163
<code>\addto</code> .....	21	<code>\delimN</code> .....	25, 143
<code>align</code> (environment) .....	60	<code>\delimR</code> .....	25, 143
<code>altlist</code> (style) .....	178	<code>description</code> (environment) .	176, 177
<code>altlistgroup</code> (style) .....	178	<code>description</code> (key) .....	43
<code>altlisthypergroup</code> (style) .....	178	<code>description</code> (option) .....	16
<code>atlong4col</code> (style) .....	184	<code>descriptionaccess</code> (key) .....	222
<code>atlong4colborder</code> (style) .....	185	<code>\DescriptionDUANewAcronymDef</code>	154
<code>atlong4colheader</code> (style) .....	184	<code>\DescriptionFootnoteNewAcronymDef</code>	152, 238
<code>atlong4colheaderborder</code> (style)	185	<code>\descriptionname</code> .....	20
<code>atlongragged4col</code> (style) .....	189	<code>\DescriptionNewAcronymDef</code>	155, 239
<code>atlongragged4colborder</code> (style)	190	<code>descriptionplural</code> (key) .....	43
<code>atlongragged4colheader</code> (style)	189	<code>descriptionpluralaccess</code> (key) .	222
<code>atlongragged4colheaderborder</code>		<code>doc package</code> .....	4, 11
(style) .....	190	<code>dua</code> (option) .....	16
<code>\altnewglossary</code> .....	42	<code>\DUANewAcronymDef</code> .....	160
<code>altsuper4col</code> (style) .....	199		
<code>altsuper4colborder</code> (style) .....	200	<b>E</b>	
<code>altsuper4colheader</code> (style) .....	200	<code>entrycounter</code> (option) .....	8
<code>altsuper4colheaderborder</code> (style)	200	<code>entrycounterwithin</code> (option) .....	8
<code>altsuperragged4col</code> (style) .....	204	<code>\entryname</code> .....	20
<code>altsuperragged4colborder</code> (style)	205	<code>environments:</code>	
<code>altsuperragged4colheader</code> (style)	205	<code>align</code> .....	60
<code>altsuperragged4colheaderborder</code>		<code>description</code> .....	176, 177
(style) .....	206	<code>longtable</code> .....	7, 165, 180–190
<code>alttree</code> (style) .....	211	<code>multicols</code> .....	191
<code>alttreegroup</code> (style) .....	213	<code>supertabular</code> ....	7, 165, 194–206
<code>alttreehypergroup</code> (style) .....	214	<code>theglossary</code> ....	4, 5, 25, 139,
<code>amsgen</code> package .....	3, 58	140, 142, 192, 193, 208, 209, 211	
<code>\andname</code> .....	20	<code>theindex</code> .....	206
<code>array</code> package .....	185, 201	<code>equation counter</code> .....	60, 61
<code>article</code> class .....	130	<code>etoolbox</code> package .....	4, 170
<b>B</b>		<b>F</b>	
<code>babel</code> package .....	18, 19, 21, 36, 242	<code>file types</code>	
		<code>.aux</code> .....	134
<b>C</b>		<code>.glo</code> .....	53
<code>\capitalisewords</code> .....	171	<code>.ist</code> .....	116, 117, 123, 124
<code>\changes</code> .....	4	<code>.toc</code> .....	28
<code>compatible-2.07</code> (option) .....	18	<code>.xdy</code> .....	23
<code>counter</code> (key) .....	45	<code>glo</code> .....	169
<code>counter</code> (option) .....	13	<code>\findrootlanguage</code> .....	35
<code>\CustomAcronymFields</code> .....	161	<code>first</code> (key) .....	44
<code>\CustomNewAcronymDef</code> .....	162	<code>firstaccess</code> (key) .....	221
		<code>\firstacronymfont</code> .....	150
<b>D</b>		<code>firstplural</code> (key) .....	44
<code>\DeclareAcronymList</code> .....	12	<code>firstpluralaccess</code> (key) .....	222
<code>\DefaultNewAcronymDef</code> ...	151, 238	<code>footnote</code> (option) .....	16
<code>\defglstdisplay</code> ....	41, 43, 44, 57, 58	<code>\FootnoteNewAcronymDef</code> ..	157, 239
<code>\defglstdisplayfirst</code>	41, 43, 44, 57, 58		

\forall glossaries .....	38	user1 .....	45
\forall glossentries .....	38	user2 .....	45
\forall glossentries .....	38	user3 .....	46
		user4 .....	46
		user5 .....	46
		user6 .....	46
<b>G</b>		glossary package .....	1, 146
\glolinkprefix .....	60	glossary styles:	
glossentry counter .....	138	altlist .....	178
glossaries package .....		altlistgroup .....	178
.....	37, 117, 164, 177, 214, 221	altlisthypergroup .....	178
glossaries-accsupp package ....	53, 221	altlong4col .....	184, 185, 189
\GlossariesWarning .....	18	altlong4colborder .....	185
\GlossariesWarningNoLine ....	18	altlong4colheader .....	184
\glossary .....	41, 124, 127, 142	altlong4colheaderborder ..	185
glossary counters:		altlongragged4col ....	189, 190
glossaryentry .....	137	altlongragged4colborder ..	190
glossarysubentry .....	137	altlongragged4colheader ..	189
glossary keys:		altlongragged4colheaderborder	
access .....	221	.....	190
counter .....	45	altsuper4col .....	199, 200, 204
description .....	43	altsuper4colborder .....	200
descriptionaccess .....	222	altsuper4colheader .....	200
descriptionplural .....	43	altsuper4colheaderborder .	200
descriptionpluralaccess ..	222	altsuperragged4col ...	204–206
first .....	44	altsuperragged4colborder .	205
firstaccess .....	221	altsuperragged4colheader .	205
firstplural .....	44	altsuperragged4colheaderborder	
firstpluralaccess .....	222	.....	206
long .....	46	alttree .....	193, 211, 213
longaccess .....	222	alttreegroup .....	213, 214
longplural .....	46	alttreehypergroup .....	214
longpluralaccess .....	222	index .....	191, 206–208
name .....	43	indexgroup .....	207, 208
nonumberlist .....	45	indexhypergroup .....	208
parent .....	45	inline .....	174
plural .....	44	list .....	6, 177–179
pluralaccess .....	222	listdotted .....	179
see .....	45	listgroup .....	177
short .....	46	listhypergroup .....	177
shortaccess .....	222	long .....	180, 181, 186
shortplural .....	46	long3col .....	181, 182
shortpluralaccess .....	222	long3colborder .....	182
sort .....	43	long3colheader .....	182
symbol .....	44	long3colheaderborder ....	182
symbolaccess .....	222	long4col .....	183, 184
symbolplural .....	44	long4colborder .....	184
symbolpluralaccess .....	222	long4colheader .....	183
text .....	43	long4colheaderborder ....	184
textaccess .....	221		
type .....	44		

longborder .....	181	treehypergroup .....	209
longheader .....	181	treenoname .....	192, 210
longheaderborder .....	181	treenonamegroup .....	210, 211
longragged .....	186, 187	treenonamehypergroup .....	211
longragged3col .....	187, 188	glossary-hypernav package .....	117
longragged3colborder .....	188	glossary-list package .....	6, 7, 176
longragged3colheader .....	188	glossary-long package .....	7, 180, 189, 194, 195
longragged3colheaderborder .....	188	glossary-longragged package .....	185
longraggedborder .....	186	glossary-mcols package .....	191
longraggedheader .....	187	glossary-super package .....	
longraggedheaderborder .....	187	.....	7, 180, 194, 201, 204
mcolalmtree .....	193	glossary-superragged package .....	201
mcolalmtreegroup .....	193, 194	glossary-tree package .....	8, 206
mcolalmtreehypergroup .....	194	glossaryentry (counter) .....	137
mcolindex .....	191	glossaryentry counter .....	8, 138, 139
mcolindexgroup .....	191	\glossaryentryfield .....	44, 140, 142, 143
mcolindexhypergroup .....	191	\glossaryentrynumber .....	137
mcoltree .....	192	\glossaryentrynumbers .....	6, 24, 133, 135
mcoltreegroup .....	192	\glossaryheader .....	140, 142
mcoltreehypergroup .....	192	\glossarymark .....	8, 26
mcoltreenoname .....	192, 193	\glossaryname .....	20, 21
mcoltreenonamegroup .....	193	\glossarypostamble .....	25, 142
mcoltreenonamehypergroup .....	193	\glossarypreamble .....	25, 142
sublistdotted .....	179	\glossarysection .....	5, 25, 41
super .....	195, 196, 202	\glossarystyle .....	142, 165
super3col .....	196, 197	glossarysubentry (counter) .....	137
super3colborder .....	197	glossarysubentry counter .....	9, 138, 139
super3colheader .....	197	\GLS .....	73
super3colheaderborder .....	197	\Gls .....	71, 75, 170
super4col .....	198, 199	\gls .....	3, 44, 56–58, 70,
super4colborder .....	199	.....	73, 74, 80, 81, 83, 84, 86, 87, 89,
super4colheader .....	198	.....	90, 92, 93, 95, 96, 98, 99, 101, 138
super4colheaderborder .....	199	\gls@Alphpage .....	128
superborder .....	195	\gls@alphpage .....	128
superheader .....	196	\gls@checkisacronymlist .....	13
superheaderborder .....	196	\gls@codepage .....	17
superragged .....	201–203	\gls@disablepagerefexpansion .....	128
superragged3col .....	203, 204	\gls@doclearpage .....	27
superragged3colborder .....	203	\gls@hypergroup prerun .....	173
superragged3colheader .....	204	\gls@level .....	47
superragged3colheaderborder .....	204	\gls@numberpage .....	128
.....	204	\gls@protected@pagefmts .....	128
superraggedborder .....	202	\gls@romanpage .....	128
superraggedheader .....	202	\gls@save@numberlist .....	132
superraggedheaderborder .....	202	\gls@suffixF .....	24
superraggedright3colheaderborder .....	204	\gls@suffixFF .....	24
.....	204	\gls@accessdisplay .....	228
tree .....	192, 208–211	\gls@sacccsupp .....	225
treegroup .....	192, 209	\gls@sadd .....	56, 115, 116, 142

<code>\glsadd options</code>	<code>\glsentryfirstaccess</code> .....	224
counter .....	<code>\Glsentryfirstplural</code> .....	111
format .....	<code>\glsentryfirstplural</code> .....	111
<code>\glsaddall</code> .....	<code>\glsentryfirstpluralaccess</code> ..	224
<code>\glsaddall options</code>	<code>\Glsentryfull</code> .....	114
types .....	<code>\glsentryfull</code> .....	114
<code>\GlsAddLetterGroup</code> .....	<code>\Glsentryfullpl</code> .....	114
<code>\GlsAddSortRule</code> .....	<code>\glsentryfullpl</code> .....	114
<code>\GlsAddXdyAlphabet</code> .....	<code>\glsentryitem</code> .....	139
<code>\GlsAddXdyAttribute</code> .....	<code>\Glsentrylong</code> .....	113
<code>\GlsAddXdyCounters</code> .....	<code>\glsentrylong</code> .....	113
<code>\GlsAddXdyLocation</code> .....	<code>\glsentrylongaccess</code> .....	225
<code>\GlsAddXdyStyle</code> .....	<code>\Glsentrylongpl</code> .....	114
<code>\glsautoprefix</code> .....	<code>\glsentrylongpl</code> .....	113
<code>\glsclearpage</code> .....	<code>\glsentrylongpluralaccess</code> ...	225
<code>\glsclosebrace</code> .....	<code>\Glsentryname</code> .....	110
<code>\glscompositor</code> .....	<code>\glsentryname</code> .....	109, 132
<code>\glscounter</code> .....	<code>\glsentrynumberlist</code> .....	114
<code>\glsdefaulttype</code> .....	<code>\Glsentryplural</code> .....	110
<code>\glsdefmain</code> .....	<code>\glsentryplural</code> .....	110
<code>\GLSdesc</code> .....	<code>\glsentrypluralaccess</code> .....	224
<code>\Glsdesc</code> .....	<code>\Glsentryshort</code> .....	113
<code>\glsdesc</code> .....	<code>\glsentryshort</code> .....	113
<code>\GLSdescplural</code> .....	<code>\glsentryshortaccess</code> .....	225
<code>\Glsdescplural</code> .....	<code>\Glsentryshortpl</code> .....	113
<code>\glsdescplural</code> .....	<code>\glsentryshortpl</code> .....	113
<code>\glsdescriptionaccessdisplay</code>	<code>\glsentryshortpluralaccess</code> ..	225
<code>\glsdescriptionpluralaccessdisplay</code>	<code>\glsentrysort</code> .....	112
.....	<code>\Glsentrysymbol</code> .....	111
<code>\glsdescwidth</code> ....	<code>\glsentrysymbol</code> .....	111
<code>\glsdisablehyper</code> .....	<code>\glsentrysymbolaccess</code> .....	225
<code>\glsdisp</code> .....	<code>\Glsentrysymbolplural</code> .....	111
<code>\glsdisplay</code> ....	<code>\glsentrysymbolplural</code> .....	111
<code>\glsdisplayfirst</code> ..	<code>\glsentrysymbolpluralaccess</code> .	225
<code>\glsdisplaynumberlist</code> .....	<code>\Glsentrytext</code> .....	110
<code>\glsdoifexists</code> .....	<code>\glsentrytext</code> .....	110, 132
<code>\glsdoifnoexists</code> .....	<code>\glsentrytextaccess</code> .....	224
<code>\glsenablehyper</code> .....	<code>\glsentrytype</code> .....	112
<code>\glsentryaccess</code> .....	<code>\Glsentryuseri</code> .....	112
<code>\glsentrycounter</code> .....	<code>\glsentryuseri</code> .....	112
<code>\glsentrycounterlabel</code> .....	<code>\Glsentryuserii</code> .....	112
<code>\Glsentrydesc</code> .....	<code>\glsentryuserii</code> .....	112
<code>\glsentrydesc</code> .....	<code>\Glsentryuseriii</code> .....	112
<code>\glsentrydescaccess</code> .....	<code>\glsentryuseriii</code> .....	112
<code>\Glsentrydescplural</code> .....	<code>\Glsentryuseriv</code> .....	112
<code>\glsentrydescplural</code> .....	<code>\glsentryuseriv</code> .....	112
<code>\glsentrydescpluralaccess</code> ...	<code>\Glsentryuserv</code> .....	113
<code>\Glsentryfirst</code> .....	<code>\glsentryuserv</code> .....	112
<code>\glsentryfirst</code> .....	<code>\Glsentryuservi</code> .....	113

<code>\glsentryuservi</code> .....	113	<code>\glsmcols</code> .....	191
<code>\GLSfirst</code> .....	82	<code>\glsmoveentry</code> .....	52
<code>\Glsfirst</code> .....	82	<code>\GLSname</code> .....	87
<code>\glsfirst</code> .....	81, 82	<code>\Glsname</code> .....	86
<code>\glsfirstaccessdisplay</code> .....	226	<code>\glsname</code> .....	86, 87
<code>\GLSfirstplural</code> .....	85	<code>\glsnameaccessdisplay</code> .....	226
<code>\Glsfirstplural</code> .....	85	<code>\glsnamefont</code> .....	143
<code>\glsfirstplural</code> .....	84, 85	<code>\glsnavhyperlink</code> .....	172
<code>\glsfirstpluralaccessdisplay</code> .....	227	<code>\glsnavhypertarget</code> .....	173
<code>\glsgetgrouplabel</code> .....	141	<code>\glsnavigation</code> .....	174
<code>\glsgetgrouptitle</code> .....	117, 141	<code>\glsnextpages</code> .....	137
<code>\glsgroupheading</code> .....	141, 142	<code>\glsnonextpages</code> .....	137
<code>\glsgroupskip</code> .....	141, 142, 177	<code>\glsnoxindywarning</code> .....	28
<code>\glshyperlink</code> .....	115	<code>\glsnumberformat</code> .....	24
<code>\glshypernavsep</code> .....	174	<code>\glsnumbersgroupname</code> ..	20, 117, 141
<code>\glshypernumber</code> .....	24, 143	<code>\glsnumlistlastsep</code> .....	115
<code>\glsIfListOfAcronyms</code> .....	12	<code>\glsnumlistsep</code> .....	115
<code>\glsinlinedescformat</code> .....	176	<code>\glsopenbrace</code> .....	117
<code>\glsinlinedopostchild</code> ...	175, 176	<code>\glsorder</code> .....	16
<code>\glsinlineemptydescformat</code> ...	176	<code>\glsorg@endtheglossary</code> .....	5
<code>\glsinlinenameformat</code> .....	176	<code>\glsorg@glossary</code> .....	4
<code>\glsinlineparentchildseparator</code> .....	176	<code>\glsorg@theglossary</code> .....	4
<code>\glsinlinepostchild</code> .....	176	<code>\glsorg@wrglossary</code> .....	4
<code>\glsinlineseparator</code> .....	176	<code>\glspagelistwidth</code> .....	180, 186, 195, 201
<code>\glsinlinesubdescformat</code> .....	176	<code>\glspar</code> .....	22
<code>\glsinlinesubnameformat</code> .....	176	<code>\GLSpl</code> .....	77
<code>\glsinlinesubseparator</code> .....	176	<code>\Glspl</code> .....	75, 170
<code>\glskeylisttok</code> .....	150	<code>\glspl</code> .....	56, 57, 74, 75, 77
<code>\glslabeltok</code> .....	150	<code>\GLSplural</code> .....	84
<code>\glslink</code> .....	56, 59, 70, 115, 142, 143	<code>\Glsplural</code> .....	83
<code>\glslink options</code>		<code>\glsplural</code> .....	83, 84
counter .....	58, 70, 170	<code>\glspluralaccessdisplay</code> .....	226
format .....	58, 70, 143	<code>\glspluralsuffix</code> .....	20, 44
hyper .....	58, 70	<code>\glspostdescription</code> .....	8
local .....	58	<code>\glspostinline</code> .....	176
<code>\glslistdottedwidth</code> .....	179	<code>\glsquote</code> .....	117
<code>\glslocalreset</code> .....	55	<code>\glsrefentry</code> .....	138
<code>\glslocalresetall</code> .....	55	<code>\glsreset</code> .....	55
<code>\glslocalunset</code> .....	55	<code>\glsresetall</code> .....	55
<code>\glslocalunsetall</code> .....	56	<code>\glsresetentrylist</code> .....	137
<code>\glslongaccessdisplay</code> .....	228	<code>\glsresetsubentrycounter</code> ....	138
<code>\glslongaccesskey</code> .....	241	<code>\glssee</code> .....	131
<code>\glslongkey</code> .....	147	<code>\glsseeformat</code> .....	119, 131
<code>\glslongpluralaccessdisplay</code> ..	228	<code>\glsseeitem</code> .....	132
<code>\glslongpluralaccesskey</code> .....	241	<code>\glsseeitemformat</code> .....	132
<code>\glslongpluralkey</code> .....	147	<code>\glsseelastsep</code> .....	132
<code>\glslongtok</code> .....	150	<code>\glsseelist</code> .....	131
<code>\glsmakefirsttuc</code> .....	171	<code>\glsseesep</code> .....	132
		<code>\glsSetAlphaCompositor</code> .....	24

<code>\glsSetCompositor</code> .....	23	<code>\GLSuseriii</code> .....	97
<code>\glsSetSuffixF</code> .....	24	<code>\Glsuseriii</code> .....	97
<code>\glsSetSuffixFF</code> .....	24	<code>\glsuseriii</code> .....	96, 97
<code>\glssettoctitle</code> .....	20	<code>\GLSuseriv</code> .....	99
<code>\glssetwidest</code> .....	211	<code>\Glsuseriv</code> .....	98
<code>\GlsSetXdyCodePage</code> .....	37	<code>\glsuseriv</code> .....	98, 99
<code>\GlsSetXdyFirstLetterAfterDigits</code> .....	117	<code>\GLSuseriv</code> .....	100
<code>\GlsSetXdyLanguage</code> .....	37	<code>\Glsuseriv</code> .....	100
<code>\GlsSetXdyLocationClassOrder</code> ..	34	<code>\glsuseriv</code> .....	99, 100
<code>\GlsSetXdyMinRangeLength</code> ....	117	<code>\GLSuservi</code> .....	102
<code>\GlsSetXdyStyles</code> .....	35	<code>\Glsuservi</code> .....	101
<code>\glsshortaccessdisplay</code> .....	228	<code>\glsuservi</code> .....	101, 102
<code>\glsshortaccesskey</code> .....	241	<code>\glswrite</code> .....	126
<code>\glsshortkey</code> .....	147	<code>\glswritefiles</code> .....	126
<code>\glsshortpluralaccessdisplay</code> ..	228		
<code>\glsshortpluralaccesskey</code> ....	241	<b>H</b>	
<code>\glsshortpluralkey</code> .....	147	<code>\hyperbf</code> .....	145
<code>\glsshorttok</code> .....	150	<code>\hyperemph</code> .....	145
<code>\glssortnumberfmt</code> .....	9	<code>hyperfirst (option)</code> .....	16
<code>\glsstepentry</code> .....	138	<code>\hyperit</code> .....	145
<code>\glsstepsubentry</code> .....	138	<code>\hyperlink</code> .....	69
<code>\glssubentrycounterlabel</code> ....	139	<code>\hypermd</code> .....	145
<code>\glssubentryitem</code> .....	139	<code>\hyperpage</code> .....	143
<code>\GLSsymbol</code> .....	91	<code>hyperref package</code> ....	130, 133, 143, 170
<code>\Glsymbol</code> .....	91	<code>\hyperrm</code> .....	145
<code>\glsymbol</code> .....	90, 91	<code>\hypersc</code> .....	145
<code>\glsymbolaccessdisplay</code> ....	227	<code>\hypersf</code> .....	145
<code>\glsymbolnav</code> .....	174	<code>\hypersl</code> .....	145
<code>\GLSsymbolplural</code> .....	93	<code>\hypertarget</code> .....	69
<code>\Glsymbolplural</code> .....	92	<code>\hypertt</code> .....	145
<code>\glsymbolplural</code> .....	92, 93	<code>\hyperup</code> .....	145
<code>\glsymbolpluralaccessdisplay</code> ..	227		
<code>\glsymbolsgroupname</code> ..	20, 117, 141	<b>I</b>	
<code>\glstarget</code> .....	140	<code>\if@gls@docloaded</code> .....	4
<code>\GLStext</code> .....	80	<code>\if@glsisacronymlist</code> .....	13
<code>\Glstext</code> .....	81	<code>\ifglossaryexists</code> .....	39
<code>\glstext</code> .....	80	<code>\ifglentryexists</code> .....	39
<code>\glstextaccessdisplay</code> .....	226	<code>\ifglshaschildren</code> .....	40
<code>\glstextformat</code> .....	56, 57	<code>\ifglshasparent</code> .....	40
<code>\glstreeindent</code> .....	209, 210	<code>\ifglsused</code> .....	39, 55
<code>\glsunset</code> .....	55	<code>\ifglxindy</code> .....	17
<code>\glsunsetall</code> .....	56	<code>index (style)</code> .....	206
<code>\GLSuseri</code> .....	94	<code>indexgroup (style)</code> .....	207
<code>\Glsuseri</code> .....	94	<code>indexhypergroup (style)</code> .....	208
<code>\glsuseri</code> .....	93, 94	<code>indexonlyfirst (option)</code> .....	15
<code>\GLSuserii</code> .....	96	<code>inline (style)</code> .....	174
<code>\Glsuserii</code> .....	95	<code>\inputencodingname</code> .....	17
<code>\glsuserii</code> .....	95, 96	<code>\istfile</code> .....	126
		<code>\istfilename</code> .....	23
		<code>\item</code> .....	143, 176, 206, 207



<b>L</b>	
\language	35
link text	56
list (style)	177
listdotted (style)	179
listgroup (style)	177
listhypergroup (style)	177
\loadglsentries	11, 56
long (key)	46
long (style)	180
long3col (style)	181
long3colborder (style)	182
long3colheader (style)	182
long3colheaderborder (style)	182
long4col (style)	183
long4colborder (style)	184
long4colheader (style)	183
long4colheaderborder (style)	184
longaccess (key)	222
longborder (style)	181
longheader (style)	181
longheaderborder (style)	181
longplural (key)	46
longpluralaccess (key)	222
longragged (style)	186
longragged3col (style)	187
longragged3colborder (style)	188
longragged3colheader (style)	188
longragged3colheaderborder (style)	188
longraggedborder (style)	186
longraggedheader (style)	187
longraggedheaderborder (style)	187
longtable (environment)	7, 165, 180–190
longtable package	180, 185
<b>M</b>	
\makefirsttuc	170
makeglossaries	16, 23, 35, 37, 41, 125, 133, 134
\makeglossaries	19, 23, 24, 40, 42, 125
\makeglossary	125
makeindex	17, 20, 23, 25, 40–43, 54, 61, 64, 116, 117, 119, 122–124, 129, 130, 140, 141, 216
delim_n	25
delim_r	25
page_compositor	23
special characters	62, 63, 117
\MakeUppercase	8
mcolalttree (style)	193
mcolalttreegroup (style)	193
mcolalttreehypergroup (style)	194
mcolindex (style)	191
mcolindexgroup (style)	191
mcolindexhypergroup (style)	191
mcoltree (style)	192
mcoltreegroup (style)	192
mcoltreehypergroup (style)	192
mcoltreename (style)	192
mcoltreenamegroup (style)	193
mcoltreenamehypergroup (style)	193
memoir class	127
mfisttuc package	1
multicol package	191
multicols (environment)	191
<b>N</b>	
name (key)	43
\newacronym	16, 46, 56, 146, 147
\newacronymhook	150
\newglossary	13, 41, 42, 124, 125, 136
\newglossaryentry	13, 47, 56, 146, 147
\newglossaryentry options	
access	223, 224
counter	45
description	13, 14, 16, 43, 47, 57, 87, 110, 147, 157, 222
descriptionaccess	225, 227
descriptionplural	89, 222
descriptionpluralaccess	225, 227
first	44, 50, 57, 70, 81, 111, 156, 159, 221
firstaccess	224, 226
firstplural	44, 50, 57, 84, 111, 222
firstpluralaccess	224, 227
format	118
long	113, 222
longaccess	225, 228
longplural	113, 222
longpluralaccess	225, 228
name	13, 14, 43, 46, 47, 86, 109, 132, 221
nonumberlist	45
parent	45, 47
plural	44, 50, 57, 83, 222
pluralaccess	224, 226

see ..... 7, 45  
 short ..... 113, 222  
 shortaccess ..... 225, 228  
 shortplural ..... 113, 222  
 shortpluralaccess ..... 225, 228  
 sort ..... 13, 15, 43, 112, 140, 141  
 symbol .....  
     13, 14, 43, 44, 57, 90, 111, 153,  
     154, 156, 159, 183, 198, 221–223  
 symbolaccess ..... 225, 227  
 symbolplural ..... 92, 222  
 symbolpluralaccess ..... 225, 227  
 text ..... 43,  
     44, 57, 70, 80, 110, 153, 156, 221  
 textaccess ..... 224, 226  
 type ..... 11, 44, 56, 111  
 user1 ..... 93, 112, 223  
 user2 ..... 95, 112  
 user3 ..... 96, 112  
 user4 ..... 98, 112  
 user5 ..... 99, 112  
 user6 ..... 101, 113, 223  
 \newglossarystyle ..... 142  
 ngerman package ..... 36  
 nogroupskip (option) ..... 8  
 \noist ..... 123, 169, 221  
 nolist (option) ..... 7  
 nolong (option) ..... 7  
 nonumberlist (key) ..... 45  
 nonumberlist (option) ..... 7  
 \nopostdesc ..... 22  
 nopostdot (option) ..... 8  
 nostyles (option) ..... 8  
 nosuper (option) ..... 7  
 notree (option) ..... 8  
 numberedsection (option) ..... 6  
 numberline (option) ..... 5

## O

\oldacronym ..... 146  
 order (option) ..... 16

## P

package options:  
   acronym .... 11, 12, 20, 135, 146, 147  
   acronym ..... 12  
   acronymlists ..... 13  
   compatible-2.07 ..... 18  
   counter ..... 13  
   counter ..... 13

description ..... 156  
 description ..... 16  
 dua ..... 155, 156  
 dua ..... 16  
 entrycounter ..... 137  
   true ..... 8  
 entrycounter ..... 8  
 entrycounterwithin ..... 8  
 footnote ..... 71–73, 75,  
     76, 78, 79, 153, 155–157, 229–234  
 footnote ..... 16  
 hyperfirst  
   false ..... 71–73, 75, 76, 78, 79  
 hyperfirst ..... 16  
 indexonlyfirst ..... 263  
 indexonlyfirst ..... 15  
 makeindex ..... 120, 170  
 nogroupskip ..... 8  
 nolist ..... 164  
 nolist ..... 7  
 nolong ..... 165, 180  
 nolong ..... 7  
 nomain ..... 11  
 nonumberlist ..... 7  
 nonumberlist ..... 7  
 nopostdot ..... 8  
 nostyles ..... 8  
 nosuper ..... 165  
 nosuper ..... 7  
 notree ..... 165  
 notree ..... 8  
 numberedsection ..... 6  
 numberline ..... 5  
 numberline ..... 5  
 order ..... 16  
 sanitize ..... 14, 15, 43, 109–111  
 sanitize ..... 15  
 savenumberlist ..... 7  
 savewrites ..... 17, 260  
   false ..... 124  
   true ..... 126  
 savewrites ..... 17  
 section ..... 5, 26  
 section ..... 5  
 seeautonumberlist ..... 7  
 shotcuts ..... 16  
 smallcaps ..... 16  
 smaller ..... 16

sort	\SetAcronymStyle	12, 161
def	\SetCustomDisplayStyle	161
standard	\SetCustomStyle	162
use	\SetDefaultAcronymDisplayStyle	
sort		150
style	\SetDefaultAcronymStyle	151
style	\SetDescriptionAcronymDisplayStyle	
subentrycounter		155
subentrycounter	\SetDescriptionAcronymStyle	156
toc	\SetDescriptionDUAAcronymDisplayStyle	
true		154
toc	\SetDescriptionDUAAcronymStyle	
translate		154
translate	\SetDescriptionFootnoteAcronymDisplayStyle	
ucmark		152
xindy	\SetDescriptionFootnoteAcronymStyle	
\pagelistname		153
parent (key)	\SetDUADisplayStyle	160
\phantomsection	\SetDUASStyle	160
plural (key)	\setentrycounter	142
pluralaccess (key)	\SetFootnoteAcronymDisplayStyle	
polyglossia package		156
\PrintChanges	\SetFootnoteAcronymStyle	157
\printglossaries	\setglossarysection	27
.. 11, 25, 40, 42, 125, 133, 135, 172	\SetSmallAcronymDisplayStyle	158
\printglossary	\SetSmallAcronymStyle	159
... 25, 40, 125, 133, 135, 136, 172	\setStyleFile	22, 23
\printglossary options	short (key)	46
nonumberlist	shortaccess (key)	222
numberedsection	shortplural (key)	46
style	shortpluralaccess (key)	222
title	shotcuts (option)	16
toctitle	\showacronymlists	168
type	\showglocounter	166
\protect	\showglodesc	167
	\showglodescaccess	242
	\showglodescplural	167
	\showglodescpluralaccess	242
	\showglofirst	166
	\showglofirstaccess	241
	\showglofirststpl	166
	\showglofirststplpluralaccess	241
	\showgloflag	168
	\showgloindex	168
	\showglolevel	165
	\showglolongaccess	242
	\showglolongpluralaccess	242
	\showgloname	167
	\showglonameaccess	241
<b>R</b>		
\renewglossarystyle		142
\roman		32
\rootlanguage		36, 37
<b>S</b>		
sanitize (option)		15
savenumberlist (option)		7
savewrites (option)		17
section (option)		5
see (key)		45
seeautonumberlist (option)		7
\seename		20
\SetAcronymLists		13



\writeist .	23, 29, 30, 34, 118, 215, 216	xindy .....	17, 23, 28, 31, 33, 35, 37, 38, 53, 54, 67, 68, 117– 119, 129, 133, 134, 140, 169, 216
<b>X</b>			
\xcapitalisewords .....	172	\xmakefirstuc .....	171
\xglaccsupp .....	226	xspace package .....	3, 146