

Documented Code For glossaries v3.03

Nicola L.C. Talbot

School of Computing Sciences
University of East Anglia
Norwich. Norfolk
NR4 7TJ. United Kingdom.

<http://theoval.cmp.uea.ac.uk/~nlct/>

2012-09-21

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 v3.03: \LaTeX 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

1	Main Package Code	3
1.1	Package Definition	3
1.2	Package Options	4
1.3	Default values	18
1.4	Xindy	27
1.5	Loops and conditionals	36
1.6	Defining new glossaries	39
1.7	Defining new entries	41
1.8	Resetting and unsetting entry flags	53
1.9	Loading files containing glossary entries	54
1.10	Using glossary entries in the text	55
1.10.1	Links to glossary entries	56
1.10.2	Displaying entry details without adding information to the glossary	105
1.11	Adding an entry to the glossary without generating text	112
1.12	Creating associated files	112
1.13	Writing information to associated files	122
1.14	Glossary Entry Cross-References	125
1.15	Displaying the glossary	127
1.16	Acronyms	140
1.17	Predefined acronym styles	144
1.18	Predefined Glossary Styles	159
1.19	Debugging Commands	159
1.20	Compatibility with version 2.07 and below	164
2	Mfirstuc Documented Code	164
3	Glossary Styles	166
3.1	Glossary hyper-navigation definitions (glossary-hypernav package)	166
3.2	In-line Style (glossary-inline.sty)	169
3.3	List Style (glossary-list.sty)	171
3.4	Glossary Styles using longtable (the glossary-long package)	174
3.5	Glossary Styles using longtable (the glossary-longragged package)	179
3.6	Glossary Styles using multicol (glossary-mcols.sty)	185
3.7	Glossary Styles using supertabular environment (glossary-super package)	188
3.8	Glossary Styles using supertabular environment (glossary-superragged package)	195
3.9	Tree Styles (glossary-tree.sty)	200
4	glossaries-compatible-207	208

5 Accessibility Support (glossaries-accsupp Code)	215
5.1 Defining Replacement Text	215
5.2 Accessing Replacement Text	218
5.3 Displaying the Glossary	231
5.4 Acronyms	231
5.5 Debugging Commands	235
6 Multi-Lingual Support	236
6.1 Babel Captions	236
6.2 Polyglossia Captions	242
6.3 Brazilian Dictionary	246
6.4 Danish Dictionary	246
6.5 Dutch Dictionary	246
6.6 English Dictionary	246
6.7 French Dictionary	247
6.8 German Dictionary	247
6.9 Irish Dictionary	247
6.10 Italian Dictionary	248
6.11 Magyar Dictionary	248
6.12 Polish Dictionary	248
6.13 Serbian Dictionary	249
6.14 Spanish Dictionary	249
Index	250

1 Main Package Code

1.1 Package Definition

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

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries}[2012/09/21 v3.03 (NLCT)]
```

Required packages:

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

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.)

```
7 \RequirePackage{amsgen}
```

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

```
8 \RequirePackage{etoolbox}
```

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.

```
9 \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`.

```
10 \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`.

```
11 \ifcsundef{chapter}%
12   {\newcommand*{\@@glossarysec}{section}}%
13   {\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`.

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

```

Determine whether or not to use numbered sections.

`\@@glossarysecstar`

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

```

`\@@glossaryseclabel`

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

```

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

```
19 \newcommand*{\glsautoprefix}{}

```

`numberedsection`

```
20 \define@choicekey{glossaries.sty}{numberedsection}{\val\nr}{%
21 false,nolabel,autolabel}[nolabel]{%
22   \ifcase\nr\relax
23     \renewcommand*{\@@glossarysecstar}{*}%
24     \renewcommand*{\@@glossaryseclabel}{}%
25   \or
26     \renewcommand*{\@@glossarysecstar}{}%
27     \renewcommand*{\@@glossaryseclabel}{}%
28   \or
29     \renewcommand*{\@@glossarysecstar}{}%
30     \renewcommand*{\@@glossaryseclabel}{}%

```

```

31 \label{\glsautoprefix\@glo@type}}%
32 \fi
33 }

```

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

`\@glossary@default@style`

```

34 \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](#).

```

35 \define@key{glossaries.sty}{style}{%
36 \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”:

`\@glossaryentrynumbers`

```

37 \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).

```

38 \DeclareOptionX{nonumberlist}{%
39 \renewcommand*\@glossaryentrynumbers[1]{\gls@save@numberlist{#1}}%
40 }

```

savenumberlist Provide means to store the number list for entries.

```

41 \define@boolkey{glossaries.sty}[gls]{savenumberlist}[true]{}
42 \gls@savenumberlistfalse

```

`\@glo@seeautonumberlist`

```

43 \newcommand*\@glo@seeautonumberlist{}

```

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

```

44 \DeclareOptionX{seeautonumberlist}{%
45 \renewcommand*\@glo@seeautonumberlist{%
46 \def\@glo@prefix{\glsnextpages}%
47 }%
48 }

```

`\@gls@loadlong`

```

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

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

```

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

```

51 \IfFileExists{supertabular.sty}{%
52   \newcommand*{\@gls@loadsuper}{\RequirePackage{glossary-super}}}{%
53   \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.

```

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

```

`\@gls@loadlist`

```

55 \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.

```

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

```

`\@gls@loadtree`

```

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

```

`notree` 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.

```

58 \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).

```

59 \DeclareOptionX{nostyles}{%
60   \renewcommand*{\@gls@loadlong}{}%
61   \renewcommand*{\@gls@loadsuper}{}%
62   \renewcommand*{\@gls@loadlist}{}%
63   \renewcommand*{\@gls@loadtree}{}%
64   \let\@glossary@default@style\relax
65 }

```

`\glspostdescription` The description terminator is given by `\glspostdescription` (except for the 3 and 4 column styles). This is a full stop by default:

```

66 \newcommand*{\glspostdescription}{\ifglsnopostdot\else.\fi}

```

nopostdot	<p>Boolean option to suppress post description dot</p> <pre>67 \define@boolkey{glossaries.sty}[gls]{nopostdot}[true]{} 68 \glsnopostdotfalse</pre>
nogroupskip	<p>Boolean option to suppress vertical space between groups in the pre-defined styles.</p> <pre>69 \define@boolkey{glossaries.sty}[gls]{nogroupskip}[true]{} 70 \glsnogroupskipfalse</pre>
ucmark	<p>Boolean option to determine whether or not to use \MakeUppercase in definition of \glossarymark</p> <pre>71 \define@boolkey{glossaries.sty}[gls]{ucmark}[true]{} 72 \glsucmarkfalse</pre>
entrycounter	<p>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.</p> <pre>73 \define@boolkey{glossaries.sty}[gls]{entrycounter}[true]{} 74 \glsentrycounterfalse</pre>
entrycounterwithin	<p>This option can be used to set a parent counter for glossaryentry. This option automatically sets entrycounter=true.</p> <pre>75 \define@key{glossaries.sty}{counterwithin}{% 76 \renewcommand*{\@gls@counterwithin}{#1}% 77 \glsentrycountertrue 78 }</pre>
\@gls@counterwithin	<p>The default value is no parent counter:</p> <pre>79 \newcommand*{\@gls@counterwithin}{}%</pre>
subentrycounter	<p>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.</p> <pre>80 \define@boolkey{glossaries.sty}[gls]{subentrycounter}[true]{} 81 \glssubentrycounterfalse</pre>
sort	<p>Define the sort method: sort=standard (default), sort=def (order of definition) or sort=use (order of use).</p> <pre>82 \define@choicekey{glossaries.sty}{sort}{standard,def,use}{% 83 \csname @gls@setupsort@#1\endcsname 84 }</pre>
@setupsort@standard	<p>Set up the macros for default sorting.</p> <pre>85 \newcommand*{\@gls@setupsort@standard}{% Store entry information when it's defined. 86 \def\do@glo@storeentry{\@glo@storeentry}% No count register required for standard sort. 87 \def\@gls@defsortcount##1{%</pre>

Sort according to sort key (\@glo@sort) if provided otherwise sort according to the entry's name (\@glo@name).

```

88 \def\@gls@defsort##1##2{%
89   \ifx\@glo@sort\@glsdefaultsort
90     \let\@glo@sort\@glo@name
91   \fi

92   \@gls@sanitizesort
93   \expandafter\protected@xdef\csname glo@##2@sort\endcsname{\@glo@sort}%
94 }%
```

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

```

95 \def\@gls@setsort##1{%
96 }
```

Set standard sort as the default:

```

97 \@gls@setupsort@standard
```

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

```

98 \newcommand*\glssortnumberfmt[1]{%
99   \ifnum#1<100000 0\fi
100  \ifnum#1<10000 0\fi
101  \ifnum#1<1000 0\fi
102  \ifnum#1<100 0\fi
103  \ifnum#1<10 0\fi
104  \number#1%
105 }
```

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

```

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

Store entry information when it's defined.

```

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

Defined count register associated with the glossary.

```

108 \def\@gls@defsortcount##1{%
109   \expandafter\global
110   \expandafter\newcount\csname glossary@##1@sortcount\endcsname
111 }%
```

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

```

112 \def\@gls@defsort##1##2{%
113   \expandafter\global\expandafter
114   \advance\csname glossary@##1@sortcount\endcsname by 1\relax
115   \expandafter\protected@xdef\csname glo@##2@sort\endcsname{%
116     \expandafter\glssortnumberfmt
117     {\csname glossary@##1@sortcount\endcsname}}%
118 }%
```


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

```
119 \def\@gls@setsort##1{%  
120 }
```

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

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

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

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

Defined count register associated with the glossary.

```
123 \def\@gls@defsortcount##1{%  
124 \expandafter\global  
125 \expandafter\newcount\csname glossary@##1@sortcount\endcsname  
126 }%
```

Initialise the sort key to empty.

```
127 \def\@gls@defsort##1##2{%  
128 \expandafter\gdef\csname glo@##2@sort\endcsname{%  
129 }%
```

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

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

Get the parent, if one exists

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

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

```
132 \ifx\@glo@parent\@empty  
133 \else  
134 \expandafter\@gls@setsort\expandafter{\@glo@parent}%  
135 \fi
```

Set index information for this entry

```
136 \edef\@glo@type{\csname glo@##1@type\endcsname}%  
137 \edef\@gls@tmp{\csname glo@##1@sort\endcsname}%  
138 \ifx\@gls@tmp\@empty  
139 \expandafter\global\expandafter  
140 \advance\csname glossary@\@glo@type @sortcount\endcsname by 1\relax  
141 \expandafter\protected@xdef\csname glo@##1@sort\endcsname{%  
142 \expandafter\glssortnumberfmt  
143 {\csname glossary@\@glo@type @sortcount\endcsname}}%  
144 \@glo@storeentry{##1}%  
145 \fi  
146 }%  
147 }
```

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

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

```

149 \newglossary{main}{gls}{glo}{\glossaryname}%
150 }

```

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`

```

151 \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`

```

152 \newcommand*{\acronymtype}{\glsdefaulttype}

```

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

```

153 \DeclareOptionX{nomain}{%
154   \let\glsdefaulttype\relax
155   \renewcommand*{\glsdefmain}{}%
156 }

```

`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.

```

157 \define@boolkey{glossaries.sty}[gls]{acronym}[true]{%
158   \DeclareAcronymList{acronym}%
159 }

```

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

```

160 \newcommand*{\@glsacronymlists}{%

```

`\@addtoacronymlists`

```

161 \newcommand*{\@addtoacronymlists}[1]{%
162   \ifx\@glsacronymlists\@empty
163     \protected@xdef\@glsacronymlists{#1}%
164   \else
165     \protected@xdef\@glsacronymlists{\@glsacronymlists,#1}%
166   \fi
167 }

```

`\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.)

```
168 \newcommand*\DeclareAcronymList}[1]{%
169   \glsIfListOfAcronyms{#1}{\@addtoacronymlists{#1}}%
170 }
```

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

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

```
171 \newcommand{\glsIfListOfAcronyms}[1]{%
172   \edef\@do@glis@islistofacronyms{%
173     \noexpand\@glis@islistofacronyms{#1}{\@glisacronymlists}}%
174   \@do@glis@islistofacronyms
175 }
```

Internal command requires label and list to be expanded:

```
176 \newcommand{\@glis@islistofacronyms}[4]{%
177   \def\glis@islistofacronyms##1,#1,##2\end@glis@islistofacronyms{%
178     \def\@before{##1}\def\@after{##2}}%
179   \glis@islistofacronyms,#2,#1,\@nil\end@glis@islistofacronyms
180   \ifx\@after\@nnil
```

Not found

```
181   #4%
182   \else
```

Found

```
183   #3%
184   \fi
185 }
```

`\if@glisacronymlist` Convenient boolean.

```
186 \newif\if@glisacronymlist
```

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

```
187 \newcommand*\@checkisacronymlist}[1]{%
188   \glsIfListOfAcronyms{#1}%
189   {\@glisacronymlisttrue}{\@glisacronymlistfalse}%
190 }
```

`\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.)

```
191 \newcommand*\SetAcronymLists}[1]{%
192   \renewcommand*\@glisacronymlists{#1}%
193 }
```

acronymlists

```
194 \define@key{glossaries.sty}{acronymlists}{%
195   \@addtoacronymlists{#1}%
196 }
```

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`

```
197 \newcommand{\glscounter}{page}
```

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

```
198 \define@key{glossaries.sty}{counter}{%
199   \renewcommand*{\glscounter}{#1}%
200 }
```

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`

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

`\@gls@sanitizename`

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

`\@gls@sanitizesymbol`

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

\@gls@sanitizesort

```
204 \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.

```
205 \define@boolkey[gls]{sanitize}{description}[true]{%
206 \ifgls@sanitize@description
207   \renewcommand*{\@gls@sanitizedesc}{\@onelevel@sanitize\@glo@desc}%
208 \else
209   \renewcommand*{\@gls@sanitizedesc}{}%
210 \fi
211 }
```

Similarly for the name key:

```
212 \define@boolkey[gls]{sanitize}{name}[true]{%
213 \ifgls@sanitize@name
214   \renewcommand*{\@gls@sanitizename}{\@onelevel@sanitize\@glo@name}%
215 \else
216   \renewcommand*{\@gls@sanitizename}{}%
217 \fi}
```

and for the symbol key:

```
218 \define@boolkey[gls]{sanitize}{symbol}[true]{%
219 \ifgls@sanitize@symbol
220   \renewcommand*{\@gls@sanitizesymbol}{%
221 \@onelevel@sanitize\@glo@symbol}%
222 \else
223   \renewcommand*{\@gls@sanitizesymbol}{}%
224 \fi}
```

and for the sort key:

```
225 \define@boolkey[gls]{sanitize}{sort}[true]{%
226 \ifgls@sanitize@sort
227   \renewcommand*{\@gls@sanitizesort}{%
228 \@onelevel@sanitize\@glo@sort}%
229 \else
230   \renewcommand*{\@gls@sanitizesort}{}%
231 \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:

```
232 \define@key{glossaries.sty}{sanitize}[description=true,symbol=true,
233 name=true]{%
234 \ifthenelse{\equal{#1}{none}}{}%
235 }
```

```

236 \renewcommand*{\@gls@sanitizedesc}{}%
237 \renewcommand*{\@gls@sanitizename}{}%
238 \renewcommand*{\@gls@sanitizesymbol}{}%
239 }%
240 {%
241 \setkeys[gls]{sanitize}{#1}}%
242 }

```

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

```

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

Set the default value:
244 \glstranslatefalse
245 \ifpackageloaded{translator}%
246 {\glstranslatetrue}%
247 {%
248 \ifpackageloaded{polyglossia}%
249 {\glstranslatetrue}%
250 {%
251 \@ifpackageloaded{babel}{\glstranslatetrue}}}%
252 }%
253 }

```

indexonlyfirst Set whether to only index on first use.

```

254 \define@boolkey{glossaries.sty}[gls]{indexonlyfirst}[true]{%
255 \glsexindexonlyfirstfalse

```

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

```

256 \define@boolkey{glossaries.sty}[gls]{hyperfirst}[true]{%
257 \glshyperfirsttrue

```

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

```

258 \define@boolkey{glossaries.sty}[glsacr]{footnote}[true]{%
259 \ifthenelse{\boolean{glsacrdescription}}{}}%
260 {\renewcommand*{\@gls@sanitizedesc}{}}%
261 }

```

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

```

262 \define@boolkey{glossaries.sty}[glsacr]{description}[true]{%
263 \renewcommand*{\@gls@sanitizesymbol}{}}%
264 }

```

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

```

265 \define@boolkey{glossaries.sty}[glsacr]{smallcaps}[true]{%
266 \renewcommand*{\@gls@sanitizesymbol}{}}%
267 }

```

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

```

268 \define@boolkey{glossaries.sty}[glsacr]{smaller}[true]{%
269   \renewcommand*{\@gls@sanitizesymbol}{}}%
270 }

```

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

```

271 \define@boolkey{glossaries.sty}[glsacr]{dua}[true]{%
272   \renewcommand*{\@gls@sanitizesymbol}{}}%
273 }

```

`shortcuts` Define acronym shortcuts.

```

274 \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.

```

275 \newcommand*{\glsorder}{word}

```

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

```

276 \newcommand*{\@glsorder}[1]{%

```

`order`

```

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

```

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

```

279 \newif\ifglxindy

```

The default is `makeindex`:

```

280 \glxindyfalse

```

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

```

281 \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 `glsnumbers` determines whether to automatically add the `glsnumbers` letter group.

```

282 \define@boolkey[glx]{xindy}{glsnumbers}[true]{%
283 \glx@xindy@glsnumberstrue

```

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

```

284 \def\@xdy@main@language{\rootlanguage}%

```

Define key to set the language

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

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

```
286 \ifcsundef{inputencodingname}{%
287   \def\glS@codepage{}}{%
288   \def\glS@codepage{\inputencodingname}
289 }
```

Define a key to set the code page.

```
290 \define@key[glS]{xindy}{codepage}{\def\glS@codepage{#1}}
```

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

```
291 \define@key{glossaries.sty}{xindy}[]{%
292   \glSxindytrue
293   \setkeys[glS]{xindy}{#1}%
294 }
```

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

```
295 \define@boolkey{glossaries.sty}[glS]{savewrites}[true]{}%
```

Set default:

```
296 \glSsavewritesfalse
```

`\GlossariesWarning` Prints a warning message.

```
297 \newcommand*\GlossariesWarning[1]{%
298   \PackageWarning{glossaries}{#1}%
299 }
```

`savewarningNoLine` Prints a warning message without the line number.

```
300 \newcommand*\GlossariesWarningNoLine[1]{%
301   \PackageWarningNoLine{glossaries}{#1}%
302 }
```

Define package option to suppress warnings

```
303 \DeclareOptionX{nowarn}{%
304   \renewcommand*\GlossariesWarning[1]{}%
305   \renewcommand*\GlossariesWarningNoLine[1]{}%
306 }
```

`compatible-2.07`

```
307 \define@boolkey{glossaries.sty}[glS]{compatible-2.07}[true]{}%
308 \csname glScompatible-2.07false\endcsname
```


Process package options:

```
309 \ProcessOptionsX
```

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

```
310 \ifglstranslate
```

```
311 \ifpackageloaded{polyglossia}%
```

```
312 {%
```

polyglossia fakes babel so need to check for polyglossia first.

```
313 }%
```

```
314 {%
```

```
315 \ifpackageloaded{babel}%
```

```
316 {%
```

```
317 \IfFileExists{translator.sty}%
```

```
318 {%
```

```
319 \RequirePackage{translator}%
```

```
320 }%
```

```
321 }%
```

```
322 }%
```

```
323 {}
```

```
324 }
```

```
325 \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}`).

```
326 \ifthenelse{\equal{\glscounter}{section}}{%
```

```
327 {%
```

```
328 \ifcsundef{chapter}{}%
```

```
329 {%
```

```
330 \let\@gls@old@chapter\@chapter
```

```
331 \def\@chapter[#1]#2{\@gls@old@chapter[{#1}]{#2}%
```

```
332 \ifcsundef{hyperdef}{\hyperdef{section}{\thesection}}}%
```

```
333 }%
```

```
334 }%
```

```
335 {}
```

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

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

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

```

337 \newcommand*{\@onlypremakeg}[1]{%
338 \ifx\@gls@onlypremakeg\@empty
339   \def\@gls@onlypremakeg{#1}%
340 \else
341   \expandafter\toks@\expandafter{\@gls@onlypremakeg}%
342   \edef\@gls@onlypremakeg{\the\toks@,\noexpand#1}%
343 \fi}

```

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

```

344 \newcommand*{\@disable@onlypremakeg}{%
345 \@for\@thiscs:=\@gls@onlypremakeg\do{%
346   \expandafter\@disable@premakecs\@thiscs%
347 }}

```

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

```

348 \newcommand*{\@disable@premakecs}[1]{%
349   \def#1{\PackageError{glossaries}{\string#1\space may only be
350     used before \string\makeglossaries}{You can't use
351     \string#1\space after \string\makeglossaries}}%
352 }

```

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`

```

353 \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`

```

354 \providecommand*\acronymname{Acronyms}

```

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

```

355 \newcommand*\glssettoctitle[1]{%
356 \def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}

```

The following commands provide text for the headers used by some of the tabular-like glossary styles. Whether or not they get used in the glossary depends on the glossary style.

`\entryname`

```

357 \providecommand*\entryname{Notation}

```

```

\descriptionname
358 \providecommand*\descriptionname{Description}

\symbolname
359 \providecommand*\symbolname{Symbol}

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

Labels for makeindex's symbol and number groups:

glssymbolsgroupname
361 \providecommand*\glssymbolsgroupname{Symbols}

glsnumbersgroupname
362 \providecommand*\glsnumbersgroupname{Numbers}

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

\seename
364 \providecommand*\seename{see}

\andname
365 \providecommand*\andname{\&}

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

dglossarytocaptions If using , \glossaryname should be defined in terms of \translate, but if ba-
bel is also loaded, it will redefine \glossaryname whenever the language is set,
so override it. (Don't use \addto as doesn't define it.)
366 \newcommand*\addglossarytocaptions[1]{%
367   \ifcsundef{captions#1}{}%
368   {%
369     \expandafter\let\expandafter\@gls@tmp\csname captions#1\endcsname
370     \expandafter\toks@\expandafter{\@gls@tmp
371       \renewcommand*\glossaryname{\translate{Glossary}}}%
372     }%
373     \expandafter\edef\csname captions#1\endcsname{\the\toks@}%
374   }%
375 }

376 \ifglstranslate

```

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

```

377 \@ifpackageloaded{translator}{%
378   \usedictionary{glossaries-dictionary}%
379   \addglossarytocaptions{portuges}%
380   \addglossarytocaptions{portuguese}%
381   \addglossarytocaptions{brazil}%
382   \addglossarytocaptions{brazilian}%
383   \addglossarytocaptions{danish}%
384   \addglossarytocaptions{dutch}%
385   \addglossarytocaptions{afrikaans}%
386   \addglossarytocaptions{english}%
387   \addglossarytocaptions{UKenglish}%
388   \addglossarytocaptions{USenglish}%
389   \addglossarytocaptions{american}%
390   \addglossarytocaptions{australian}%
391   \addglossarytocaptions{british}%
392   \addglossarytocaptions{canadian}%
393   \addglossarytocaptions{newzealand}%
394   \addglossarytocaptions{french}%
395   \addglossarytocaptions{frenchb}%
396   \addglossarytocaptions{francais}%
397   \addglossarytocaptions{acadian}%
398   \addglossarytocaptions{canadien}%
399   \addglossarytocaptions{german}%
400   \addglossarytocaptions{germanb}%
401   \addglossarytocaptions{austrian}%
402   \addglossarytocaptions{naustrian}%
403   \addglossarytocaptions{ngerman}%
404   \addglossarytocaptions{irish}%
405   \addglossarytocaptions{italian}%
406   \addglossarytocaptions{magyar}%
407   \addglossarytocaptions{hungarian}%
408   \addglossarytocaptions{polish}%
409   \addglossarytocaptions{spanish}%
410   \renewcommand*{\glstttitle}[1]{%
411     \ifthenelse{\equal{#1}{main}}{%
412       \translatelet{\glossarytoctitle}{Glossary}}{%
413       \ifthenelse{\equal{#1}{acronym}}{%
414         \translatelet{\glossarytoctitle}{Acronyms}}{%
415         \def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}}%
416   \renewcommand*{\glossaryname}{\translate{Glossary}}%
417   \renewcommand*{\acronymname}{\translate{Acronyms}}%
418   \renewcommand*{\entryname}{\translate{Notation (glossaries)}}%
419   \renewcommand*{\descriptionname}{%
420     \translate{Description (glossaries)}}%
421   \renewcommand*{\symbolname}{\translate{Symbol (glossaries)}}%
422   \renewcommand*{\pagelistname}{%
423     \translate{Page List (glossaries)}}%
424   \renewcommand*{\glssymbolsgroupname}{%

```

```

425     \translate{Symbols (glossaries)}}}%
426     \renewcommand*{\glsnumbersgroupname}{%
427     \translate{Numbers (glossaries)}}}%
428 }{%

429     \@ifpackageloaded{polyglossia}%
430     {\RequirePackage{glossaries-polyglossia}}%
431     {%
432     \@ifpackageloaded{babel}{%
433     \RequirePackage{glossaries-babel}}}%
434 }}
435 \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.

```
436 \newcommand*{\nopostdesc}{}

```

`\@nopostdesc` Suppress next description terminator.

```

437 \newcommand*{\@nopostdesc}{%
438   \let\org@glspostdescription\glspostdescription
439   \def\glspostdescription{%
440     \let\glspostdescription\org@glspostdescription}%
441 }

```

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

```
442 \newcommand{\glspar}{\par}

```

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

```

443 \ifglxindy
444   \newcommand{\setStyleFile}[1]{%
445     \renewcommand{\istfilename}{#1.xdy}}
446 \else
447   \newcommand{\setStyleFile}[1]{%
448     \renewcommand{\istfilename}{#1.ist}}
449 \fi

```

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

```
450 \@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`

```

451 \ifglxindy
452   \def\istfilename{\jobname.xdy}
453 \else
454   \def\istfilename{\jobname.ist}
455 \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`

```
456 \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`

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

```

`\glsSetCompositor` Sets the compositor.

```
458 \newcommand*\glsSetCompositor[1]{%
459   \renewcommand*\glscompositor{#1}}

```

Only use before `\makeglossaries`

```
460 \@onlypremakeg\glsSetCompositor

```

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

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

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

```

`\glsSetAlphaCompositor` Sets the alpha compositor.

```
462 \ifglsxindy
463   \newcommand*\glsSetAlphaCompositor[1]{%
464     \renewcommand*\@glsAlphacompositor{#1}}
465 \else
466   \newcommand*\glsSetAlphaCompositor[1]{%
467     \glsnoxywarning\glsSetAlphaCompositor}
468 \fi

```

Can only be used before `\makeglossaries`

```
469 \@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.

```
470 \newcommand*\gls@suffiXF{}

```

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

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

Only has an effect when used before `\makeglossaries`

```
473 \@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.

```
474 \newcommand*\gls@suffixFF{-}
```

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

```
475 \newcommand*\glsSetSuffixFF}[1]{%  
476   \renewcommand*\gls@suffixFF{#1}%  
477 }
```

`\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”.

```
478 \ifcsundef{hyperlink}%  
479 {%  
480   \newcommand*\glsnumberformat}[1]{#1}%  
481 }%  
482 {%  
483   \newcommand*\glsnumberformat}[1]{\glshypernumber{#1}}%  
484 }
```

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`

```
485 \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`

```
486 \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 `\glossary preamble`.) 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 `\glossary preamble` before each `\printglossary`.

`\glossary preamble`

```
487 \newcommand*{\glossary preamble}{}

```

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

```
\renewcommand{\glossary postamble}{For a complete list of terms
see \cite{blah}\gdef\glossary preamble{}}
```

`\glossary postamble`

```
488 \newcommand*{\glossary postamble}{}

```

`\glossary section` The sectioning command that starts a glossary is given by `\glossary section`. (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@glossary section`, otherwise it uses `\@glossary section`.

```
489 \newcommand*{\glossary section}[2][\@gls@title]{%
490   \def\@gls@title{#2}%
491   \ifcsundef{phantomsection}%
492   {%
493     \@glossary section{#1}{#2}%
494   }%
495   {%
496     \p@glossary section{#1}{#2}%
497   }%
498   \glossary mark{\glossary to ctitle}%
499 }
```

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

```
500 \ifcsundef{glossary mark}%
501 {%
502   \ifglsucmark
503     \newcommand{\glossary mark}[1]{%
504       \@mkboth{\MakeUppercase{#1}}{\MakeUppercase{#1}}%
505     }
506   \else
507     \newcommand{\glossary mark}[1]{\@mkboth{#1}{#1}}
```



```

508 \fi
509 }%
510 {%
511 \GlossariesWarning{overriding \string\glossarymark}%
512 \@ifclassloaded{memoir}%
513 {
514 \ifglsucmark
515 \renewcommand{\glossarymark}[1]{%
516 \mkboth{\MakeUppercase{#1}}{\MakeUppercase{#1}}%
517 }
518 \else
519 \renewcommand{\glossarymark}[1]{%
520 \markboth{\memUHead{#1}}{\memUHead{#1}}%
521 }
522 \fi
523 }
524 {
525 \ifglsucmark
526 \renewcommand{\glossarymark}[1]{%
527 \mkboth{\MakeUppercase{#1}}{\MakeUppercase{#1}}%
528 }
529 \else
530 \renewcommand{\glossarymark}[1]{\mkboth{#1}{#1}}
531 \fi
532 }
533 }

```

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`

```

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

```

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

`\@glossarysection`

```

536 \newcommand*\@glossarysection[2]{%
537 \ifx\@glossarysecstar\@empty
538 \csname\@glossarysec\endcsname{#2}%
539 \else
540 \csname\@glossarysec\endcsname*{#2}%
541 \@gls@toc{#1}{\@glossarysec}%
542 \fi
543 \@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.

`\@p@glossarysection`

```
544 \newcommand*\@p@glossarysection}[2]{%
545 \glsclearpage
546 \phantomsection
547 \ifx\@glossarysecstar\@empty
548   \csname\@glossarysec\endcsname{#2}%
549 \else
550   \@gls@toc{#1}{\@glossarysec}%
551   \csname\@glossarysec\endcsname*{#2}%
552 \fi
553 \@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.

```
554 \newcommand*\gls@doclearpage{%
555   \ifthenelse{\equal{\@glossarysec}{chapter}}{
556     {
557       \ifcsundef{cleardoublepage}{\clearpage}{\cleardoublepage}%
558     }%
559   }%
560 }
```

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

```
561 \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`

```
562 \newcommand*\@gls@toc}[2]{%
563 \ifglstoc
564   \ifglslnumberline
565     \addcontentsline{toc}{#2}{\numberline{#1}}%
566   \else
567     \addcontentsline{toc}{#2}{#1}%
568   \fi
569 \fi}
```

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

```
570 \newcommand*{\glsnoxindywarning}[1]{%
571   \GlossariesWarning{Not in xindy mode --- ignoring \string#1}%
572 }
```

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

```
573 \ifglsxindy
574   \edef\@xdyattributes{\string"default\string"}%
575 \fi
```

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

```
576 \ifglsxindy
577   \edef\@xdyattributelist{}%
578 \fi
```

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

```
579 \ifglsxindy
580   \def\@xdylocref{}
581 \fi
```

`\@gls@ifinlist`

```
582 \newcommand*{\@gls@ifinlist}[4]{%
583   \def\@do@ifinlist##1,#1,##2\end@do@ifinlist{%
584     \def\@gls@listsuffix{##2}%
585     \ifx\@gls@listsuffix\@empty
586       #4%
587     \else
588       #3%
589     \fi
590   }%
591   \@do@ifinlist,#2,#1,\end@do@ifinlist
592 }
```

`\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.

```
593 \ifglsxindy
594   \newcommand*{\@xdycounters}{\glscounter}
595   \newcommand*\GlsAddXdyCounters[1]{%
596     \@for\@gls@ctr:=#1\do{%
```

Check if already in list before adding.

```

597     \edef\@do@addcounter{%
598         \noexpand\@gls@ifinlist{\@gls@ctr}{\@xdycounters}{}%
599         {%
600             \noexpand\edef\noexpand\@xdycounters{\@xdycounters,%
601                 \noexpand\@gls@ctr}%
602         }%
603     }%
604     \@do@addcounter
605 }
606 }

```

Only has an effect before `\writeist`:

```

607 \@onlypremakeg\GlsAddXdyCounters
608 \else
609 \newcommand*\GlsAddXdyCounters[1]{%
610     \glsnoxindywarning\GlsAddXdyAttribute
611 }
612 \fi

```

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

```

613 \newcommand*\@disabled@glssaddxdycounters{%
614     \PackageError{glossaries}{\string\GlsAddXdyCounters\space
615         can't be used after \string\GlsAddXdyAttribute}{Move all
616         occurrences of \string\GlsAddXdyCounters\space before the first
617         instance of \string\GlsAddXdyAttribute}%
618 }

```

`\GlsAddXdyAttribute` Adds an attribute.

```

619 \ifglsxindy

```

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

```

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

```

Add to xindy attribute list

```

621     \edef\@xdyattributes{\@xdyattributes ^^J \string"#1\string" ^^J
622         \string"#2#1\string"}%

```

Add to xindy markup location.

```

623     \expandafter\toks@\expandafter{\@xdylocref}%
624     \edef\@xdylocref{\the\toks@ ^^J%
625         (markup-locref
626         :open \string"\string~n%
627         \expandafter\string\csname glsX#2X#1\endcsname
628         \string" ^^J
629         :close \string"\string" ^^J
630         :attr \string"#2#1\string")}%

```

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

```

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

```

```

632     \setentrycounter[##1]{#2}\csname #1\endcsname{##2}%
633   }%
634 }

```

High-level command:

```

635 \newcommand*\GlsAddXdyAttribute[1]{%

```

Add to comma-separated attribute list

```

636   \ifx\@xdyattributelist\@empty
637     \edef\@xdyattributelist{#1}%
638   \else
639     \edef\@xdyattributelist{\@xdyattributelist,#1}%
640   \fi

```

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

```

641   \@for\@this@counter:=\@xdycounters\do{%
642     \protected@edef\gls@do@addxdyattribute{%
643       \noexpand\@glsaddxdyattribute{#1}{\@this@counter}%
644     }
645     \gls@do@addxdyattribute
646   }%

```

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

```

647   \let\GlsAddXdyCounters\@disabled@glsaddxdycounters
648 }

```

Only has an effect before `\writeist`:

```

649 \@onlypremakeg\GlsAddXdyAttribute
650 \else
651   \newcommand*\GlsAddXdyAttribute[1]{%
652     \glsnoxindywarning\GlsAddXdyAttribute}
653 \fi

```

redefinedattributes Add known attributes for all defined counters

```

654 \ifglxsindy
655 \newcommand*{\@gls@addpredefinedattributes}{%
656   \GlsAddXdyAttribute{glsnumberformat}
657   \GlsAddXdyAttribute{textrm}
658   \GlsAddXdyAttribute{textsf}
659   \GlsAddXdyAttribute{texttt}
660   \GlsAddXdyAttribute{textbf}
661   \GlsAddXdyAttribute{textmd}
662   \GlsAddXdyAttribute{textit}
663   \GlsAddXdyAttribute{textup}
664   \GlsAddXdyAttribute{textsl}
665   \GlsAddXdyAttribute{textsc}
666   \GlsAddXdyAttribute{emph}
667   \GlsAddXdyAttribute{glshypernumber}
668   \GlsAddXdyAttribute{hyperrrm}
669   \GlsAddXdyAttribute{hypersf}
670   \GlsAddXdyAttribute{hypertt}

```

```

671 \GlsAddXdyAttribute{hyperbf}
672 \GlsAddXdyAttribute{hypermd}
673 \GlsAddXdyAttribute{hyperit}
674 \GlsAddXdyAttribute{hyperup}
675 \GlsAddXdyAttribute{hypersl}
676 \GlsAddXdyAttribute{hypersc}
677 \GlsAddXdyAttribute{hyperemph}
678 }
679 \else
680 \let\@gls@addpredefinedattributes\relax
681 \fi

```

`\@xdyuseralphabets` List of additional alphabets

```

682 \def\@xdyuseralphabets{}

```

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

```

683 \ifglsxindy
684 \newcommand*{\GlsAddXdyAlphabet}[2]{%
685 \edef\@xdyuseralphabets{%
686 \@xdyuseralphabets ^^J
687 (define-alphabet "#1" (#2))}}
688 \else
689 \newcommand*{\GlsAddXdyAlphabet}[2]{%
690 \glsnoxindywarning\GlsAddXdyAlphabet}
691 \fi

```

This code is only required for xindy:

```

692 \ifglsxindy

```

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

```

693 \newcommand*{\@gls@xdy@locationlist}{%
694 roman-page-numbers,%
695 Roman-page-numbers,%
696 arabic-page-numbers,%
697 alpha-page-numbers,%
698 Alpha-page-numbers,%
699 Appendix-page-numbers,%
700 arabic-section-numbers%
701 }

```

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.

```

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

```

```

703     \string"roman-numbers-lowercase\string" :sep \string"}}%
704 \@onelevel@sanitize\@gls@roman
705 \edef\@tmp{\string" \string"roman-numbers-lowercase\string"
706     :sep \string"}}%
707 \@onelevel@sanitize\@tmp
708 \ifx\@tmp\@gls@roman
709     \expandafter
710     \edef\csname @gls@xdy@Lclass@roman-page-numbers\endcsname{%
711         \string"roman-numbers-lowercase\string"%
712     }%
713 \else
714     \expandafter
715     \edef\csname @gls@xdy@Lclass@roman-page-numbers\endcsname{
716         :sep \string"\@gls@roman\string"%
717     }%
718 \fi

```

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

```

719 \expandafter\def\csname @gls@xdy@Lclass@Roman-page-numbers\endcsname{%
720     \string"roman-numbers-uppercase\string"%
721 }%

```

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

```

722 \expandafter\def\csname @gls@xdy@Lclass@arabic-page-numbers\endcsname{%
723     \string"arabic-numbers\string"%
724 }%

```

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

```

725 \expandafter\def\csname @gls@xdy@Lclass@alpha-page-numbers\endcsname{%
726     \string"alpha\string"%
727 }%

```

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

```

728 \expandafter\def\csname @gls@xdy@Lclass@Alpha-page-numbers\endcsname{%
729     \string"ALPHA\string"%
730 }%

```

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

```

731 \expandafter\def\csname @gls@xdy@Lclass@Appendix-page-numbers\endcsname{%
732     \string"ALPHA\string"
733     :sep \string"\@glsAlphacompositor\string"
734     \string"arabic-numbers\string"%
735 }

```

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

```

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

```

```

737 \string"arabic-numbers\string"
738 :sep \string"\glscompositor\string"
739 \string"arabic-numbers\string"%
740 }%

```

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

```

741 \def\xdyuserlocationdefs{}

```

`\xdyuserlocationnames` List of additional user location names

```

742 \def\xdyuserlocationnames{}

```

End of xindy-only block:

```

743 \fi

```

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

```

744 \ifglsxindy
745 \newcommand*\GlsAddXdyLocation[3][{}]{%
746 \def\@gls@tmp{#1}%
747 \ifx\@gls@tmp\@empty
748 \edef\xdyuserlocationdefs{%
749 \xdyuserlocationdefs ^^J%
750 (define-location-class \string"#2\string"^^J\space\space
751 \space(:sep \string"{}\glsopenbrace\string" #3
752 :sep \string"\glsclosebrace\string"))
753 }%
754 \else
755 \edef\xdyuserlocationdefs{%
756 \xdyuserlocationdefs ^^J%
757 (define-location-class \string"#2\string"^^J\space\space
758 \space(:sep "\glsopenbrace"
759 #1
760 :sep "\glsclosebrace\glsopenbrace" #3
761 :sep "\glsclosebrace"))
762 }%
763 \fi
764 \edef\xdyuserlocationnames{%
765 \xdyuserlocationnames^^J\space\space\space
766 \string"#1\string"}%
767 }

```

Only has an effect before `\writeist`:

```

768 \@onlypremake\GlsAddXdyLocation
769 \else
770 \newcommand*\GlsAddXdyLocation[2]{%
771 \glsnoxindywarning\GlsAddXdyLocation}
772 \fi

```


ylocationclassorder Define location class order

```
773 \ifglxindy
774   \edef\@xdylocationclassorder{^^J\space\space\space
775     \string"roman-page-numbers\string"^^J\space\space\space
776     \string"arabic-page-numbers\string"^^J\space\space\space
777     \string"arabic-section-numbers\string"^^J\space\space\space
778     \string"alpha-page-numbers\string"^^J\space\space\space
779     \string"Roman-page-numbers\string"^^J\space\space\space
780     \string"Alpha-page-numbers\string"^^J\space\space\space
781     \string"Appendix-page-numbers\string"
782     \@xdyuserlocationnames^^J\space\space\space
783     \string"see\string"
784   }
785 \fi
```

Change the location order.

yLocationClassOrder

```
786 \ifglxindy
787   \newcommand*\GlsSetXdyLocationClassOrder[1]{%
788     \def\@xdylocationclassorder{#1}}
789 \else
790   \newcommand*\GlsSetXdyLocationClassOrder[1]{%
791     \glsnxindywarning\GlsSetXdyLocationClassOrder}
792 \fi
```

\@xdysortrules Define sort rules

```
793 \ifglxindy
794   \def\@xdysortrules{}
795 \fi
```

\GlsAddSortRule Add a sort rule

```
796 \ifglxindy
797   \newcommand*\GlsAddSortRule[2]{%
798     \expandafter\toks@\expandafter{\@xdysortrules}%
799     \protected@edef\@xdysortrules{\the\toks@ ^^J
800       (sort-rule \string"#1\string" \string"#2\string")}%
801   }
802 \else
803   \newcommand*\GlsAddSortRule[2]{%
804     \glsnxindywarning\GlsAddSortRule}
805 \fi
```

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

```
806 \ifglxindy
807   \def\@xdyrequiredstyles{tex}
808 \fi
```

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

```
809 \ifglxindy
810   \newcommand*\GlsAddXdyStyle[1]{%
811     \edef\@xdyrequiredstyles{\@xdyrequiredstyles,#1}}%
812 \else
813   \newcommand*\GlsAddXdyStyle[1]{%
814     \glsnxindywarning\GlsAddXdyStyle}
815 \fi
```

`\GlsSetXdyStyles` Reset the list of required styles

```
816 \ifglxindy
817   \newcommand*\GlsSetXdyStyles[1]{%
818     \edef\@xdyrequiredstyles{#1}}
819 \else
820   \newcommand*\GlsSetXdyStyles[1]{%
821     \glsnxindywarning\GlsSetXdyStyles}
822 \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.

```
823 \ifglxindy
824   \@ifpackageloaded{babel}{%

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

825   \def\findrootlanguage{\begingroup
826     \escapechar=-1\relax

normalize \language to category 12 chars

827     \edef\language{\%
828       \expandafter\string\csname\language\endcsname}%

disable babel.sty useless commands

829     \def\NeedsTeXFormat##1[##2]{}%
830     \def\ProvidesPackage##1[##2]{}%
831     \let\LdfInit\relax
832     \def\languageattribute##1##2{}%

change the meaning of \DeclareOption

833     \def\DeclareOption##1##2{%
      at \DeclareOption* we end

834       \ifx##1*\expandafter\endinput\else
        else we build a string with the first argument

835       \edef\testlanguage{\expandafter\string\csname##1\endcsname}%
```

if \testlanguage and \language are the same we execute the second argument

```
836 \ifx\testlanguage\language##2\fi
837 \fi}
```

almost all options of babel are \input{<name>.ldf}

```
838 \def\input##1{\stripldf##1}%
```

we put the root language name in \rootlanguage

```
839 \def\stripldf##1.ldf{\gdef\rootlanguage{##1}}%
```

now input babel.sty, using the primitive \input

```
840 \@@input babel.sty
841 \endgroup}%
842 }{%
```

hasn't been loaded, so check if has been loaded

```
843 \ifpackageloaded{ngerman}{%
844 \def\findrootlanguage{%
845 \def\rootlanguage{german}}%
846 }{%
```

Neither babel nor ngerman have been loaded, so assume the root language is English

```
847 \def\findrootlanguage{%
848 \def\rootlanguage{english}}%
849 }%
850 }%
851 \fi
```

\rootlanguage Set default root language to English.

```
852 \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.

```
853 \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.

```
854 \ifglxindy
855 \newcommand*\GlsSetXdyLanguage[2][\glsdefaulttype]{%
856 \ifglossaryexists{#1}{%
857 \expandafter\def\csname @xdy@#1@language\endcsname{#2}%
858 }{%
859 \PackageError{glossaries}{Can't set language type for
860 glossary type '#1' --- no such glossary}{%
861 You have specified a glossary type that doesn't exist}}}
```

```

862 \else
863   \newcommand*\GlsSetXdyLanguage[2] [] {%
864     \glsnoxindywarning\GlsSetXdyLanguage}
865 \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.

```

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

```

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

```

867 \ifglxindy
868   \newcommand*\GlsSetXdyCodePage[1] {%
869     \renewcommand*\@gls@codepage{#1}%
870   }
871 \else
872   \newcommand*\GlsSetXdyCodePage[1] {%
873     \glsnoxindywarning\GlsSetXdyCodePage}
874 \fi

```

`\@xdylettergroups` Store letter group definitions.

```

875 \ifglxindy
876   \ifgls@xindy@glsnumbers
877     \def\@xdylettergroups{(define-letter-group
878       \string"glsnumbers\string"^^J\space\space\space
879       :prefixes (\string"0\string" \string"1\string"
880       \string"2\string" \string"3\string" \string"4\string"
881       \string"5\string" \string"6\string" \string"7\string"
882       \string"8\string" \string"9\string")^^J\space\space\space
883       :before \string"@glsfirstletter\string")}
884   \else
885     \def\@xdylettergroups{}
886   \fi
887 \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.

```

888   \newcommand*\GlsAddLetterGroup[2] {%
889     \expandafter\toks@\expandafter{\@xdylettergroups}%
890     \protected@edef\@xdylettergroups{\the\toks@^^J%
891     (define-letter-group \string"#1\string"^^J\space\space\space#2)}%
892   }%

```

1.5 Loops and conditionals

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

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

where `⟨cmd⟩` is a control sequence which will be set to the name of the glossary in the current iteration.

```
893 \newcommand*\forallglossaries[3][\@glo@types]{%
894   \@for#2:=#1\do{\ifx#2\@empty\else#3\fi}%
895 }
```

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

`\forall glossentries [⟨type⟩] {⟨cmd⟩} {⟨code⟩}`

where `⟨type⟩` is the glossary label and `⟨cmd⟩` is a control sequence which will be set to the entry label in the current iteration.

```
896 \newcommand*\forallglossentries[3][\glsdefaulttype]{%
897   \edef\@glo@list{\csname glolist@#1\endcsname}%
898   \@for#2:=\@glo@list\do{\ifx#2\@empty\else#3\fi}%
899 }
```

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

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

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

```
900 \newcommand*\forallglossentries[3][\@glo@types]{%
901   \expandafter\forallglossaries\expandafter[#1]{\@this@glo@}%
902   \forallglossentries[\@this@glo@]{#2}{#3}}
```

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

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

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

```
903 \newcommand{\ifglossaryexists}[3]{%
904   \ifcsundef{glo@#1@out}{#3}{#2}%
905 }
```

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

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

where `⟨label⟩` is the entry's label.

```
906 \newcommand{\ifglossentryexists}[3]{%
907   \ifcsundef{glo@#1@name}{#3}{#2}%
908 }
```

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

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

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

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

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

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

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

```
910 \newcommand{\glsoifexists}[2]{%
911   \ifglstryexists{#1}{#2}{%
912     \PackageError{glossaries}{Glossary entry ‘#1’ has not been
913       defined}{You need to define a glossary entry before you
914       can use it.}}%
915 }
```

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

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

```
916 \newcommand{\glsoifnoexists}[2]{%
917   \ifglstryexists{#1}{%
918     \PackageError{glossaries}{Glossary entry ‘#1’ has already
919       been defined}{}}{#2}%
920 }
```

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

```
921 \newcommand{\ifglshaschildren}[3]{%
922   \glsoifexists{#1}%
923   {%
924     \def\do@glshaschildren{#3}%
925     \expandafter\forlstryentries\expandafter[\csname glo@#1@type\endcsname]
926     {\glo@label}%
927     {%
928       \letcs\glo@parent{glo@\glo@label @parent}%
929       \ifthenelse{\equal{#1}{\glo@parent}}{%
930         {%
931           \def\do@glshaschildren{#2}%
932           \@endfortrue
933         }%
934         {}%
935       }%
936       \do@glshaschildren
937     }%
938 }
```

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

```

939 \newcommand{\ifglshasparent}[3]{%
940   \glsdoifexists{#1}%
941   {%
942     \ifcsemt{glo@#1@parent}{#3}{#2}%
943   }%
944 }

```

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`

```

945 \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`

```

946 \newcommand*{\newglossary}[5][glg]{%
947   \ifglossaryexists{#2}{%
948     \PackageError{glossaries}{Glossary type ‘#2’ already exists}{%
949       You can’t define a new glossary called ‘#2’ because it already
950       exists}%
951   }{%

```

Check if default has been set

```

952   \ifx\glsdefaulttype\relax
953     \gdef\glsdefaulttype{#2}%
954   \fi

```

Add this to the list of glossary types:

```

955   \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.

```

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

```

Store details of this new glossary type:

```

957 \expandafter\def\csname @glotype@#2@in\endcsname{#3}%
958 \expandafter\def\csname @glotype@#2@out\endcsname{#4}%
959 \expandafter\def\csname @glotype@#2@title\endcsname{#5}%
960 \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.

```

961 \ifcsundef{gls@#2@display}%
962 {%
963   \expandafter\gdef\csname gls@#2@display\endcsname{\glsdisplay}%
964 }%
965 }%
966 \ifcsundef{gls@#2@displayfirst}%
967 {%
968   \expandafter\gdef\csname gls@#2@displayfirst\endcsname{%
969     \glsdisplayfirst
970   }%
971 }%
972 }%

```

Define sort counter if required:

```

973 \@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.)

```

974 \@ifnextchar[{\@gls@setcounter{#2}}%
975   {\@gls@setcounter{#2}[\glscounter]}}%
976 }

```

`\altnewglossary`

```

977 \newcommand*\altnewglossary}[3]{%
978   \newglossary[#2-glg]{#1}{#2-gls}{#2-glo}{#3}%
979 }

```

Only define new glossaries in the preamble:

```

980 \@onlypreamble{\newglossary}

```

Only define new glossaries before `\makeglossaries`

```

981 \@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`

```

982 \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`

```
983 \def\@gls@setcounter#1[#2]{%
984   \expandafter\def\csname @gls@#1@counter\endcsname{#2}%
    Add counter to xindy list, if not already added:
985   \ifglsxindy
986     \GlsAddXdyCounters{#2}%
987   \fi
988 }
```

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

`\@gls@getcounter`

```
989 \newcommand*\@gls@getcounter[1]{%
990   \csname @gls@#1@counter\endcsname}
```

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

```
991 \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.

```
992 \define@key{glossentry}{name}{%
993   \def\@glo@name{#1}%
994 }
```

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.)

```
995 \define@key{glossentry}{description}{%  
996 \def\@glo@desc{#1}%  
997 }
```

descriptionplural

```
998 \define@key{glossentry}{descriptionplural}{%  
999 \def\@glo@descplural{#1}%  
1000 }
```

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

```
1001 \define@key{glossentry}{sort}{%  
1002 \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.

```
1003 \define@key{glossentry}{text}{%  
1004 \def\@glo@text{#1}%  
1005 }
```

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.

```
1006 \define@key{glossentry}{plural}{%  
1007 \def\@glo@plural{#1}%  
1008 }
```

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.

```
1009 \define@key{glossentry}{first}{%  
1010 \def\@glo@first{#1}%  
1011 }
```

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.

```
1012 \define@key{glossentry}{firstplural}{%  
1013 \def\@glo@firstplural{#1}%  
1014 }
```

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).

```
1015 \define@key{glossentry}{symbol}{%
1016 \def\@glo@symbol{#1}%
1017 }
```

symbolplural

```
1018 \define@key{glossentry}{symbolplural}{%
1019 \def\@glo@symbolplural{#1}%
1020 }
```

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

```
1021 \define@key{glossentry}{type}{%
1022 \def\@glo@type{#1}}
```

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

```
1023 \define@key{glossentry}{counter}{%
1024 \ifcsundef{c@#1}%
1025 {%
1026 \PackageError{glossaries}%
1027 {There is no counter called ‘#1’}%
1028 {%
1029 The counter key should have the name of a valid counter
1030 as its value%
1031 }%
1032 }%
1033 {%
1034 \def\@glo@counter{#1}%
1035 }%
1036 }
```

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

```
1037 \define@key{glossentry}{see}{%
1038 \def\@glo@see{#1}%
1039 \@glo@seeautonumberlist
1040 }
```

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

```
1041 \define@key{glossentry}{parent}{%
1042 \def\@glo@parent{#1}}
```

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

```
1043 \define@choicekey{glossentry}{nonumberlist}[\val\nr]{true,false}[true]{%
1044   \ifcase\nr\relax
1045   \def\@glo@prefix{\glslonextpages}%
1046   \else
1047   \def\@glo@prefix{\glsnextpages}%
1048   \fi
1049 }
```

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

user1

```
1050 \define@key{glossentry}{user1}{%
1051   \def\@glo@useri{#1}%
1052 }
```

user2

```
1053 \define@key{glossentry}{user2}{%
1054   \def\@glo@userii{#1}%
1055 }
```

user3

```
1056 \define@key{glossentry}{user3}{%
1057   \def\@glo@useriii{#1}%
1058 }
```

user4

```
1059 \define@key{glossentry}{user4}{%
1060   \def\@glo@useriv{#1}%
1061 }
```

user5

```
1062 \define@key{glossentry}{user5}{%
1063   \def\@glo@userv{#1}%
1064 }
```

user6

```
1065 \define@key{glossentry}{user6}{%
1066   \def\@glo@uservi{#1}%
1067 }
```

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.

```
1068 \define@key{glossentry}{short}{%
1069   \def\@glo@short{#1}%
1070 }
```

shortplural This key is provided for use by \newacronym.

```
1071 \define@key{glossentry}{shortplural}{%
1072   \def\@glo@shortpl{#1}%
1073 }
```

long This key is provided for use by \newacronym.

```
1074 \define@key{glossentry}{long}{%
1075   \def\@glo@long{#1}%
1076 }
```

longplural This key is provided for use by \newacronym.

```
1077 \define@key{glossentry}{longplural}{%
1078   \def\@glo@longpl{#1}%
1079 }
```

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

```
1080 \newcommand*\@glsnoname{%
1081   \PackageError{glossaries}{name key required in
1082     \string\newglossaryentry\space for entry '\@glo@label'}{You
1083     haven't specified the entry name}}
```

\@glsdefaultplural Define command to set default plural.

```
1084 \newcommand*\@glsdefaultplural{\@glo@text\glspluralsuffix}
```

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

```
1085 \newcommand*\@glsmissingnumberlist[1]{%
1086   ??%
1087   \ifglssavenumberlist
1088     \GlossariesWarning{Missing number list for entry '#1'.
1089       Maybe makeglossaries + rerun required.}%
1090   \else
1091     \PackageError{glossaries}%
1092       {Package option 'savenumberlist=true' required.}%
1093     {%
1094       You must use the 'savenumberlist' package option
1095       to reference location lists.%
1096     }%
1097   \fi
1098 }
```

\@glsdefaultsort Define command to set default sort.

```
1099 \newcommand*\@glsdefaultsort{\@glo@name}
```

\gls@level Register to increment entry levels.

```
1100 \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.)

```
1101 \newrobustcmd{\newglossaryentry}[2]{%
```

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

```
1102 \glsdoifnoexists{#1}%  
1103 {%
```

Store label

```
1104 \def\@glo@label{#1}%
```

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

```
1105 \let\@glo@name\@glsnoname  
  
1106 \def\@glo@desc{%  
1107 \PackageError{glossaries}  
1108 {%  
1109 description key required in \string\newglossaryentry\space  
1110 for entry '\@glo@label'%  
1111 }%  
1112 {%  
1113 You haven't specified the entry description%  
1114 }%  
1115 }%  
  
1116 \def\@glo@descplural{\@glo@desc}%  
1117 \def\@glo@type{\@glsdefaulttype}%  
1118 \def\@glo@symbol{\relax}%  
  
1119 \def\@glo@symbolplural{\@glo@symbol}%  
1120 \def\@glo@text{\@glo@name}%  
1121 \let\@glo@plural\@glsdefaultplural
```

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

```
1122 \let\@glo@first\relax  
1123 \let\@glo@firstplural\relax
```

Set the default sort:

```
1124 \let\@glo@sort\@glsdefaultsort
```

Set the default counter:

```
1125 \def\@glo@counter{\@gls@getcounter{\@glo@type}}%  
1126 \def\@glo@see{}%  
1127 \def\@glo@parent{}%  
1128 \def\@glo@prefix{}%
```

```

1129 \def\@glo@useri{}%
1130 \def\@glo@userii{}%
1131 \def\@glo@useriii{}%
1132 \def\@glo@useriv{}%
1133 \def\@glo@userv{}%
1134 \def\@glo@uservi{}%

1135 \def\@glo@short{}%
1136 \def\@glo@shortpl{}%
1137 \def\@glo@long{}%
1138 \def\@glo@longpl{}%

```

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

```
1139 \@newglossaryentryprehook
```

Extract key-val information from third parameter:

```
1140 \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.

```

1141 \ifcsundef{glolist@\@glo@type}%
1142 {%
1143   \PackageError{glossaries}%
1144     {Glossary type '\@glo@type' has not been defined}%
1145     {You need to define a new glossary type, before making entries
1146      in it}%
1147 }%
1148 {%
1149   \protected@edef\@glolist@{\csname glolist@\@glo@type\endcsname}%
1150   \expandafter\xdef\csname glolist@\@glo@type\endcsname{\@glolist@{#1},}%
1151 }%

```

Initialise level to 0.

```
1152 \gls@level=0\relax
```

Has this entry been assigned a parent?

```
1153 \ifx\@glo@parent\@empty
```

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

```

1154 \expandafter\gdef\csname glo@#1@parent\endcsname{}%
1155 \else

```

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

```

1156 \ifthenelse{\equal{#1}{\@glo@parent}}%
1157 {%
1158   \PackageError{glossaries}{Entry '#1' can't be its own parent}{}%
1159   \def\@glo@parent{}%
1160   \expandafter\gdef\csname glo@#1@parent\endcsname{}%
1161 }%
1162 {%

```

Check the parent exists:

```
1163      \ifglentryexists{\@glo@parent}%
1164      {%
```

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

```
1165      \expandafter\xdef\csname glo@#1@parent\endcsname{\@glo@parent}%
```

Determine level.

```
1166      \gls@level=\csname glo@\@glo@parent @level\endcsname\relax
1167      \advance\gls@level by 1\relax
```

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

```
1168      \ifx\@glo@name\@glsnoname
1169      \expandafter\let\expandafter\@glo@name
1170      \csname glo@\@glo@parent @name\endcsname
```

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

```
1171      \ifx\@glo@plural\@glsdefaultplural
1172      \expandafter\let\expandafter\@glo@plural
1173      \csname glo@\@glo@parent @plural\endcsname
1174      \fi
1175      \fi
1176      }%
1177      {%
```

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

```
1178      \PackageError{glossaries}%
1179      {%
1180      Invalid parent '@glo@parent'
1181      for entry '#1' - parent doesn't exist%
1182      }%
1183      {%
1184      Parent entries must be defined before their children%
1185      }%
1186      \def\@glo@parent{%
1187      \expandafter\gdef\csname glo@#1@parent\endcsname{%
1188      }%
1189      }%
1190      \fi
```

Set the level for this entry

```
1191      \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.)

```
1192      \ifx\relax\@glo@firstplural
1193      \ifx\relax\@glo@first
```



```

1194         \def\@glo@firstplural{\@glo@plural}%
1195         \def\@glo@first{\@glo@text}%
1196     \else
1197         \def\@glo@firstplural{\@glo@first\glspluralsuffix}%
1198     \fi
1199 \else
1200     \ifx\relax\@glo@first
1201         \def\@glo@first{\@glo@text}%
1202     \fi
1203 \fi

```

Define commands associated with this entry:

```

1204 \expandafter
1205 \protected@xdef\csname glo@#1@text\endcsname{\@glo@text}%
1206 \expandafter
1207 \protected@xdef\csname glo@#1@plural\endcsname{\@glo@plural}%
1208 \expandafter
1209 \protected@xdef\csname glo@#1@first\endcsname{\@glo@first}%
1210 \expandafter
1211 \protected@xdef\csname glo@#1@firstpl\endcsname{\@glo@firstplural}%
1212 \expandafter
1213 \protected@xdef\csname glo@#1@type\endcsname{\@glo@type}%
1214 \expandafter
1215 \protected@xdef\csname glo@#1@counter\endcsname{\@glo@counter}%
1216 \expandafter
1217 \protected@xdef\csname glo@#1@useri\endcsname{\@glo@useri}%
1218 \expandafter
1219 \protected@xdef\csname glo@#1@userii\endcsname{\@glo@userii}%
1220 \expandafter
1221 \protected@xdef\csname glo@#1@useriii\endcsname{\@glo@useriii}%
1222 \expandafter
1223 \protected@xdef\csname glo@#1@useriv\endcsname{\@glo@useriv}%
1224 \expandafter
1225 \protected@xdef\csname glo@#1@userv\endcsname{\@glo@userv}%
1226 \expandafter
1227 \protected@xdef\csname glo@#1@uservi\endcsname{\@glo@uservi}%
1228 \expandafter
1229 \protected@xdef\csname glo@#1@short\endcsname{\@glo@short}%
1230 \expandafter
1231 \protected@xdef\csname glo@#1@shortpl\endcsname{\@glo@shortpl}%
1232 \expandafter
1233 \protected@xdef\csname glo@#1@long\endcsname{\@glo@long}%
1234 \expandafter
1235 \protected@xdef\csname glo@#1@longpl\endcsname{\@glo@longpl}%
1236 \@gls@sanitizename
1237 \expandafter\protected@xdef\csname glo@#1@name\endcsname{\@glo@name}%

```

Set default numberlist if not defined:

```

1238 \ifcsundef{glo@#1@numberlist}%
1239 {%

```

```

1240     \csxdef{glo@#1@numberlist}{\noexpand\@gls@missingnumberlist{\@glo@label}}%
1241 }%
1242 {}%

```

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.

```

1243     \def\@glo@@desc{\@glo@first}%
1244     \ifx\@glo@desc\@glo@@desc
1245         \let\@glo@desc\@glo@first
1246     \fi
1247     \@gls@sanitizedesc
1248     \expandafter\protected@xdef\csname glo@#1@desc\endcsname{\@glo@desc}%
1249     \expandafter\protected@xdef\csname glo@#1@descplural\endcsname{\@glo@descplural}%

```

Set the sort key for this entry:

```

1250     \@gls@defsort{\@glo@type}{#1}%

1251     \def\@glo@@symbol{\@glo@text}%
1252     \ifx\@glo@symbol\@glo@@symbol
1253         \let\@glo@symbol\@glo@text
1254     \fi
1255     \@gls@sanitizesymbol
1256     \expandafter\protected@xdef\csname glo@#1@symbol\endcsname{\@glo@symbol}%
1257     \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):

```

1258     \expandafter\gdef\csname glo@#1@flagfalse\endcsname{%
1259         \expandafter\global\expandafter
1260             \let\csname ifglo@#1@flag\endcsname\iffalse
1261     }%
1262     \expandafter\gdef\csname glo@#1@flagtrue\endcsname{%
1263         \expandafter\global\expandafter
1264             \let\csname ifglo@#1@flag\endcsname\iftrue
1265     }%
1266     \csname glo@#1@flagfalse\endcsname

```

Sort out any cross-referencing if required.

```

1267     \ifx\@glo@see\@empty
1268     \else
1269         \protected@edef\@do@glsssee{%
1270             \noexpand\@gls@fixbraces\noexpand\@glo@list\@glo@see
1271             \noexpand\@nil
1272             \noexpand\expandafter\noexpand\@glsssee\noexpand\@glo@list{#1}}%
1273         \@do@glsssee
1274     \fi
1275 }%

```

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

```

1276     \do@glo@storeentry{#1}%

```

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

```
1277 \newglossaryentryposthook
1278 }
```

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

```
1279 \newcommand*{\@newglossaryentryprehook}{}%
```

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

```
1280 \newcommand*{\@newglossaryentryposthook}{}%
```

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

```
1281 \newcommand*{\glsmoveentry}[2]{%
1282   \edef\glo@type{\csname glo@#1@type\endcsname}%
1283   \def\glo@list{,}%
1284   \for\glseentries[\glo@type]{\glo@label}%
1285   {%
1286     \ifthenelse{\equal{\glo@label}{#1}}{}{\eappto\glo@list{\glo@label,}}%
1287   }%
1288   \cslet\glolist@\glo@type{\glo@list}%
1289   \csdef{glo@#1@type}{#2}%
1290 }
```

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

```
1291 \ifglxindy
1292   \newcommand*{\@glossaryentryfield}{\string\@glossaryentryfield}
1293 \else
1294   \newcommand*{\@glossaryentryfield}{\string\glossaryentryfield}
1295 \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.)

```
1296 \ifglxindy
1297   \newcommand*{\@glossarysubentryfield}{%
1298     \string\@glossarysubentryfield}
1299 \else
1300   \newcommand*{\@glossarysubentryfield}{%
1301     \string\glossarysubentryfield}
1302 \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.)

```
1303 \newcommand{\@glo@storeentry}[1]{%
```

Get the sort string and escape any special characters

```
1304 \protected@edef\@glo@sort{\csname glo@#1@sort\endcsname}%  
1305 \@gls@checkmkidxchars\@glo@sort
```

Same again for the name string.

```
1306 \protected@edef\@glo@name{\csname glo@#1@name\endcsname}%  
1307 \@gls@checkmkidxchars\@glo@name
```

Add the font command. (The backslash needs to be escaped for xindy.)

```
1308 \ifglxindy  
1309 \protected@edef\@glo@name{\string\@glsnamefont{\@glo@name}}%  
1310 \else  
1311 \protected@edef\@glo@name{\string\glsnamefont{\@glo@name}}%  
1312 \fi
```

Get the description string and escape any special characters

```
1313 \protected@edef\@glo@desc{\csname glo@#1@desc\endcsname}%  
1314 \@gls@checkmkidxchars\@glo@desc
```

Same again for the symbol

```
1315 \protected@edef\@glo@symbol{\csname glo@#1@symbol\endcsname}%  
1316 \@gls@checkmkidxchars\@glo@symbol
```

Escape any special characters in the prefix

```
1317 \@gls@checkmkidxchars\@glo@prefix
```

Get the parent, if one exists

```
1318 \edef\@glo@parent{\csname glo@#1@parent\endcsname}%
```

Write the information to the glossary file.

```
1319 \ifglxindy
```

Store using xindy syntax.

```
1320 \ifx\@glo@parent\@empty
```

Entry doesn't have a parent

```
1321 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%  
1322 (\string\@glo@sort\string" %  
1323 \string\@glo@prefix@glossaryentryfield{#1}\@glo@name  
1324 }\@glo@desc}\@glo@symbol}\string") %  
1325 }%  
1326 \else
```

Entry has a parent

```
1327 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%  
1328 \csname glo@\@glo@parent @index\endcsname  
1329 (\string\@glo@sort\string" %  
1330 \string\@glo@prefix@glossarysubentryfield%  
1331 {\csname glo@#1@level\endcsname}{#1}\@glo@name  
1332 }\@glo@desc}\@glo@symbol}\string") %  
1333 }%  
1334 \fi  
1335 \else
```

Store using makeindex syntax.

```

1336 \ifx\@glo@parent\@empty
    Sanitize \@glo@prefix
1337     \@onelevel@sanitize\@glo@prefix
    Entry doesn't have a parent
1338     \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
1339         \@glo@sort\@gls@actualchar\@glo@prefix
1340         \@glossaryentryfield{#1}\@glo@name}\@glo@desc
1341     }\@glo@symbol}%
1342 }%
1343 \else
    Entry has a parent
1344     \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
1345         \csname glo@\@glo@parent @index\endcsname\@gls@levelchar
1346         \@glo@sort\@gls@actualchar\@glo@prefix
1347         \@glossarysubentryfield
1348         {\csname glo@#1@level\endcsname}\@glo@name}\@glo@desc
1349     }\@glo@symbol}%
1350 }%
1351 \fi
1352 \fi
1353 }

```

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
1354 \newcommand*\glsreset}[1]{%
1355 \glsdoifexists{#1}{%
1356 \expandafter\global\csname glo@#1@flagfalse\endcsname}}

```

As above, but with only a local effect:

```

\glslocalreset
1357 \newcommand*\glslocalreset}[1]{%
1358 \glsdoifexists{#1}{%
1359 \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`

```
1360 \newcommand*{\glsunset}[1]{%
1361 \glsdoifexists{#1}{%
1362 \expandafter\global\csname glo@#1@flagtrue\endcsname}}
```

As above, but with only a local effect:

`\glslocalunset`

```
1363 \newcommand*{\glslocalunset}[1]{%
1364 \glsdoifexists{#1}{%
1365 \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`

```
1366 \newcommand*{\glsresetall}[1][\@glo@types]{%
1367 \forallglsentries[#1]{\@glsentry}{%
1368 \glsreset{\@glsentry}}}
```

As above, but with only a local effect:

`\glslocalresetall`

```
1369 \newcommand*{\glslocalresetall}[1][\@glo@types]{%
1370 \forallglsentries[#1]{\@glsentry}{%
1371 \glslocalreset{\@glsentry}}}
```

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

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

`\glsunsetall`

```
1372 \newcommand*{\glsunsetall}[1][\@glo@types]{%
1373 \forallglsentries[#1]{\@glsentry}{%
1374 \glsunset{\@glsentry}}}
```

As above, but with only a local effect:

`\glslocalunsetall`

```
1375 \newcommand*{\glslocalunsetall}[1][\@glo@types]{%
1376 \forallglsentries[#1]{\@glsentry}{%
1377 \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.¹

`\loadglsentries[⟨type⟩]{⟨filename⟩}`

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

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`, `\glsp1` and uppercase variants or `\glsadd` and `\glsaddall` will appear in the glossary). The mandatory argument is the filename (with or without `.tex` extension).

`\loadglsentries`

```
1378 \newcommand*\loadglsentries[2][\@gls@default]{%
1379 \let\@gls@default\glsdefaulttype
1380 \def\glsdefaulttype{#1}\input{#2}%
1381 \let\glsdefaulttype\@gls@default}
```

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

```
1382 \@onlypreamble{\loadglsentries}
```

1.10 Using glossary entries in the text

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

`\glstextformat`

```
1383 \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 `\glsp1`. The default is to display the first parameter followed by the additional text.

`\glsdisplayfirst`

```
1384 \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 `\glsp1`.

`\glsdisplay`

```
1385 \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`

```
1386 \newcommand*{\defglsdisplay}[2][\glsdefaulttype]{%
1387 \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`

```
1388 \newcommand*{\defglsdisplayfirst}[2][\glsdefaulttype]{%
1389 \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 L^AT_EX 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.

```
1390 \define@key{glslink}{counter}{%
1391   \ifcsundef{c@#1}%
1392   {%
1393     \PackageError{glossaries}%
1394     {There is no counter called ‘#1’}%
1395     {%
1396       The counter key should have the name of a valid counter
```



```

1397         as its value%
1398     }%
1399 }%
1400 {%
1401     \def\@gls@counter{#1}%
1402 }%
1403 }

```

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.

```

1404 \define@key{glslink}{format}{%
1405 \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.

```

1406 \define@boolkey{glslink}{hyper}[true]{}

```

Syntax:

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

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

There is also a starred version:

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

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

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

```
\glslink
```

```

1407 \newrobustcmd*{\glslink}{%
1408 \@ifstar\@sgls@link\@gls@@link}

```

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

```

1409 \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.

```

1410 \newcommand*{\@gls@@link}[3] [] {%
1411     \ifglstryexists{#2}%
1412     {%
1413         \@gls@link[#1]{#2}{#3}%

```

```

1414 }{%
1415   \PackageError{glossaries}{Glossary entry ‘#2’ has not been
1416     defined}{You need to define a glossary entry before you
1417     can use it.}%

```

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

```

1418   \glstextformat{#3}%
1419 }%
1420 }

```

`\@gls@link`

```

1421 \def\@gls@link[#1]#2#3{%

```

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

```

1422   \leavevmode
1423   \def\glslabel{#2}%
1424   \def\@glsnumberformat{glsnumberformat}%
1425   \edef\@gls@counter{\csname glo@#2@counter\endcsname}%
1426   \KV@glslink@hypertrue
1427   \setkeys{glslink}{#1}%

```

Store the entry's counter in `\theglsentrycounter`

```

1428   \@gls@saveentrycounter

```

Define sort key if necessary:

```

1429   \@gls@setsort{#2}%

1430   \@do@wrglossary{#2}%
1431   \ifKV@glslink@hyper
1432     \@glslink{\glolinkprefix#2}{\glstextformat{#3}}%
1433   \else
1434     \glstextformat{#3}\relax
1435   \fi
1436 }

```

`\glolinkprefix`

```

1437 \newcommand*{\glolinkprefix}{glo:}

```

`\glsentrycounter` Set default value of entry counter

```

1438 \def\glsentrycounter{\glscounter}%

```

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

```

1439 \newcommand*{\@gls@saveentrycounter}{%
1440   \def\@gls@Hcounter{%
      Are we using equation counter?
1441     \ifthenelse{\equal{\@gls@counter}{equation}}{%
1442       {

```

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

```

1443 \ifcsundef{xatlevel@}%
1444 {%
1445 \edef\theglentrycounter{\expandafter\noexpand
1446 \csname the\@gls@counter\endcsname}%
1447 }%
1448 {%
1449 \ifx\xatlevel@\@empty
1450 \edef\theglentrycounter{\expandafter\noexpand
1451 \csname the\@gls@counter\endcsname}%
1452 \else
1453 \savecounters@
1454 \advance\c@equation by 1\relax
1455 \edef\theglentrycounter{\csname the\@gls@counter\endcsname}%

```

Check if hyperref version of this counter

```

1456 \ifcsundef{theH\@gls@counter}%
1457 {%
1458 \def\@gls@Hcounter{\theglentrycounter}%
1459 }%
1460 {%
1461 \def\@gls@Hcounter{\csname theH\@gls@counter\endcsname}%
1462 }%
1463 \protected@edef\theHglentrycounter{\@gls@Hcounter}%
1464 \restorecounters@
1465 \fi
1466 }%
1467 }%
1468 {%

```

Not using equation counter so no special measures:

```

1469 \edef\theglentrycounter{\expandafter\noexpand
1470 \csname the\@gls@counter\endcsname}%
1471 }%

```

Check if hyperref version of this counter

```

1472 \ifx\@gls@Hcounter\@empty
1473 \ifcsundef{theH\@gls@counter}%
1474 {%
1475 \def\theHglentrycounter{\theglentrycounter}%
1476 }%
1477 {%
1478 \protected@edef\theHglentrycounter{\expandafter\noexpand
1479 \csname theH\@gls@counter\endcsname}%
1480 }%
1481 \fi
1482 }

```

`\@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.

```

1483 \def\@set@glo@numformat#1#2#3#4{%
1484   \expandafter\@glo@check@mkidxrangechar#3\@nil
1485   \protected@edef#1{%
1486     \@glo@prefix setentrycounter[#4]{#2}%
1487     \expandafter\string\csname\@glo@suffix\endcsname
1488   }%
1489   \@gls@checkmkidxchars#1%
1490 }

```

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.

```

1491 \def\@glo@check@mkidxrangechar#1#2\@nil{%
1492   \if#1(\relax
1493     \def\@glo@prefix{(%}
1494     \if\relax#2\relax
1495       \def\@glo@suffix{glsnumberformat}%
1496     \else
1497       \def\@glo@suffix{#2}%
1498     \fi
1499   \else
1500     \if#1)\relax
1501       \def\@glo@prefix{)%}
1502     \if\relax#2\relax
1503       \def\@glo@suffix{glsnumberformat}%
1504     \else
1505       \def\@glo@suffix{#2}%
1506     \fi
1507   \else
1508     \def\@glo@prefix{}\def\@glo@suffix{#1#2}%
1509   \fi
1510 }

```

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

```

1511 \newcommand*\@gls@escbsdq[1]{%
1512   \def\@gls@checkedmkidx{%}
1513   \let\gls@xdystring=#1\relax
1514   \@onelevel@sanitize\gls@xdystring
1515   \edef\do@gls@xdycheckbackslash{%
1516     \noexpand\@gls@xdycheckbackslash\gls@xdystring\noexpand\@nil
1517     \@backslashchar\@backslashchar\noexpand\null}%

```

```

1518 \do@gl@xdycheckbackslash
1519 \expandafter\@gl@updatechecked\@gl@checkedmkidx{\gl@xdystring}%
1520 \def\@gl@checkedmkidx{}%
1521 \expandafter\@gl@xdycheckquote\gl@xdystring\@nil""\null
1522 \expandafter\@gl@updatechecked\@gl@checkedmkidx{\gl@xdystring}%
1523 \let#1=\gl@xdystring
1524 }

```

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

`\gl@checkmkidxchars`

```

1525 \newcommand{\@gl@checkmkidxchars}[1]{%
1526 \ifgl@xindy
1527 \@gl@escbsdq{#1}%
1528 \else
1529 \def\@gl@checkedmkidx{}%
1530 \expandafter\@gl@checkquote#1\@nil""\null
1531 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
1532 \def\@gl@checkedmkidx{}%
1533 \expandafter\@gl@checkescquote#1\@nil\""\null
1534 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
1535 \def\@gl@checkedmkidx{}%
1536 \expandafter\@gl@checkescactual#1\@nil\?\?\null
1537 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
1538 \def\@gl@checkedmkidx{}%
1539 \expandafter\@gl@checkactual#1\@nil??\null
1540 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
1541 \def\@gl@checkedmkidx{}%
1542 \expandafter\@gl@checkbar#1\@nil||\null
1543 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
1544 \def\@gl@checkedmkidx{}%
1545 \expandafter\@gl@checkescbar#1\@nil\\|\null
1546 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
1547 \def\@gl@checkedmkidx{}%
1548 \expandafter\@gl@checklevel#1\@nil!!\null
1549 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
1550 \fi
1551 }

```

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

`\@gl@updatechecked`

```

1552 \def\@gl@updatechecked#1\@nil#2{\def#2{#1}}

```

`\@gl@tmpb` Define temporary token

```

1553 \newtoks\@gl@tmpb

```

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

```

1554 \def\@gl@checkquote#1"#2"#3\null{%
1555 \@gl@tmpb=\expandafter{\@gl@checkedmkidx}%

```

```

1556 \toks@={#1}%
1557 \ifx\null#2\null
1558 \ifx\null#3\null
1559 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1560 \def\@gls@checkquote{\relax}%
1561 \else
1562 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1563 \@gls@quotechar\@gls@quotechar\@gls@quotechar\@gls@quotechar}%
1564 \def\@gls@checkquote{\@gls@checkquote#3\null}%
1565 \fi
1566 \else
1567 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1568 \@gls@quotechar\@gls@quotechar}%
1569 \ifx\null#3\null
1570 \def\@gls@checkquote{\@gls@checkquote#2""\null}%
1571 \else
1572 \def\@gls@checkquote{\@gls@checkquote#2"#3\null}%
1573 \fi
1574 \fi
1575 \@@gls@checkquote}

```

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

```

1576 \def\@gls@checkescquote#1\"#2\"#3\null{%
1577 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1578 \toks@={#1}%
1579 \ifx\null#2\null
1580 \ifx\null#3\null
1581 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1582 \def\@gls@checkescquote{\relax}%
1583 \else
1584 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1585 \@gls@quotechar\string\" \@gls@quotechar
1586 \@gls@quotechar\string\" \@gls@quotechar}%
1587 \def\@gls@checkescquote{\@gls@checkescquote#3\null}%
1588 \fi
1589 \else
1590 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1591 \@gls@quotechar\string\" \@gls@quotechar}%
1592 \ifx\null#3\null
1593 \def\@gls@checkescquote{\@gls@checkescquote#2\"\" \null}%
1594 \else
1595 \def\@gls@checkescquote{\@gls@checkescquote#2\"#3\null}%
1596 \fi
1597 \fi
1598 \@@gls@checkescquote}

```

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

```

1599 \def\@gls@checkescactual#1\?#2\?#3\null{%
1600 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%

```

```

1601 \toks@={#1}%
1602 \ifx\null#2\null
1603 \ifx\null#3\null
1604 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1605 \def\@@gls@checkescactual{\relax}%
1606 \else
1607 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1608 \@gls@quotechar\string"\@gls@actualchar
1609 \@gls@quotechar\string"\@gls@actualchar}%
1610 \def\@@gls@checkescactual{\@gls@checkescactual#3\null}%
1611 \fi
1612 \else
1613 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1614 \@gls@quotechar\string"\@gls@actualchar}%
1615 \ifx\null#3\null
1616 \def\@@gls@checkescactual{\@gls@checkescactual#2\?\?\null}%
1617 \else
1618 \def\@@gls@checkescactual{\@gls@checkescactual#2\?#3\null}%
1619 \fi
1620 \fi
1621 \@@gls@checkescactual}

```

\@gls@checkescbar Similarly for \|:

```

1622 \def\@gls@checkescbar#1\|#2\|#3\null{%
1623 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1624 \toks@={#1}%
1625 \ifx\null#2\null
1626 \ifx\null#3\null
1627 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1628 \def\@@gls@checkescbar{\relax}%
1629 \else
1630 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1631 \@gls@quotechar\string"\@gls@encapchar
1632 \@gls@quotechar\string"\@gls@encapchar}%
1633 \def\@@gls@checkescbar{\@gls@checkescbar#3\null}%
1634 \fi
1635 \else
1636 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1637 \@gls@quotechar\string"\@gls@encapchar}%
1638 \ifx\null#3\null
1639 \def\@@gls@checkescbar{\@gls@checkescbar#2\|\|\null}%
1640 \else
1641 \def\@@gls@checkescbar{\@gls@checkescbar#2\|#3\null}%
1642 \fi
1643 \fi
1644 \@@gls@checkescbar}

```

\@gls@checkesclevel Similarly for \!:

```

1645 \def\@gls@checkesclevel#1\!#2\!#3\null{%

```

```

1646 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1647 \toks@={#1}%
1648 \ifx\null#2\null
1649 \ifx\null#3\null
1650 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1651 \def\@gls@checkesclevel{\relax}%
1652 \else
1653 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1654   \@gls@quotechar\string"\@gls@levelchar
1655   \@gls@quotechar\string"\@gls@levelchar}%
1656 \def\@gls@checkesclevel{\@gls@checkesclevel#3\null}%
1657 \fi
1658 \else
1659 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1660   \@gls@quotechar\string"\@gls@levelchar}%
1661 \ifx\null#3\null
1662 \def\@gls@checkesclevel{\@gls@checkesclevel#2\!\!\null}%
1663 \else
1664 \def\@gls@checkesclevel{\@gls@checkesclevel#2\!#3\null}%
1665 \fi
1666 \fi
1667 \@gls@checkesclevel}

```

\@gls@checkbar and for |:

```

1668 \def\@gls@checkbar#1|#2|#3\null{%
1669 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1670 \toks@={#1}%
1671 \ifx\null#2\null
1672 \ifx\null#3\null
1673 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1674 \def\@gls@checkbar{\relax}%
1675 \else
1676 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1677   \@gls@quotechar\@gls@encapchar\@gls@quotechar\@gls@encapchar}%
1678 \def\@gls@checkbar{\@gls@checkbar#3\null}%
1679 \fi
1680 \else
1681 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1682   \@gls@quotechar\@gls@encapchar}%
1683 \ifx\null#3\null
1684 \def\@gls@checkbar{\@gls@checkbar#2||\null}%
1685 \else
1686 \def\@gls@checkbar{\@gls@checkbar#2|#3\null}%
1687 \fi
1688 \fi
1689 \@gls@checkbar}

```

\@gls@checklevel and for !:

```

1690 \def\@gls@checklevel#1!#2!#3\null{%

```



```

1691 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1692 \toks@={#1}%
1693 \ifx\null#2\null
1694 \ifx\null#3\null
1695 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1696 \def\@gls@checklevel{\relax}%
1697 \else
1698 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1699 \@gls@quotechar\@gls@levelchar\@gls@quotechar\@gls@levelchar}%
1700 \def\@gls@checklevel{\@gls@checklevel#3\null}%
1701 \fi
1702 \else
1703 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1704 \@gls@quotechar\@gls@levelchar}%
1705 \ifx\null#3\null
1706 \def\@gls@checklevel{\@gls@checklevel#2!!\null}%
1707 \else
1708 \def\@gls@checklevel{\@gls@checklevel#2!#3\null}%
1709 \fi
1710 \fi
1711 \@gls@checklevel}

```

\@gls@checkactual and for ?:

```

1712 \def\@gls@checkactual#1?#2?#3\null{%
1713 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1714 \toks@={#1}%
1715 \ifx\null#2\null
1716 \ifx\null#3\null
1717 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1718 \def\@gls@checkactual{\relax}%
1719 \else
1720 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1721 \@gls@quotechar\@gls@actualchar\@gls@quotechar\@gls@actualchar}%
1722 \def\@gls@checkactual{\@gls@checkactual#3\null}%
1723 \fi
1724 \else
1725 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1726 \@gls@quotechar\@gls@actualchar}%
1727 \ifx\null#3\null
1728 \def\@gls@checkactual{\@gls@checkactual#2??\null}%
1729 \else
1730 \def\@gls@checkactual{\@gls@checkactual#2?#3\null}%
1731 \fi
1732 \fi
1733 \@gls@checkactual}

```

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

```

1734 \def\@gls@xdycheckquote#1"#2"#3\null{%
1735 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%

```

```

1736 \toks@={#1}%
1737 \ifx\null#2\null
1738 \ifx\null#3\null
1739 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1740 \def\@gls@xdycheckquote{\relax}%
1741 \else
1742 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1743 \string"\string"}%
1744 \def\@gls@xdycheckquote{\@gls@xdycheckquote#3\null}%
1745 \fi
1746 \else
1747 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1748 \string"}%
1749 \ifx\null#3\null
1750 \def\@gls@xdycheckquote{\@gls@xdycheckquote#2""\null}%
1751 \else
1752 \def\@gls@xdycheckquote{\@gls@xdycheckquote#2"#3\null}%
1753 \fi
1754 \fi
1755 \@gls@xdycheckquote
1756 }

```

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

```

1757 \edef\def\@gls@xdycheckbackslash{%
1758 \noexpand\def\noexpand\@gls@xdycheckbackslash##1\@backslashchar
1759 ##2\@backslashchar##3\noexpand\null{%
1760 \noexpand\@gls@tmpb=\noexpand\expandafter
1761 {\noexpand\@gls@checkedmkidx}%
1762 \noexpand\toks@={##1}%
1763 \noexpand\ifx\noexpand\null##2\noexpand\null
1764 \noexpand\ifx\noexpand\null##3\noexpand\null
1765 \noexpand\edef\noexpand\@gls@checkedmkidx{%
1766 \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@}%
1767 \noexpand\def\noexpand\@gls@xdycheckbackslash{\relax}%
1768 \noexpand\else
1769 \noexpand\edef\noexpand\@gls@checkedmkidx{%
1770 \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
1771 \@backslashchar\@backslashchar\@backslashchar\@backslashchar}%
1772 \noexpand\def\noexpand\@gls@xdycheckbackslash{%
1773 \noexpand\@gls@xdycheckbackslash##3\noexpand\null}%
1774 \noexpand\fi
1775 \noexpand\else
1776 \noexpand\edef\noexpand\@gls@checkedmkidx{%
1777 \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
1778 \@backslashchar\@backslashchar}%
1779 \noexpand\ifx\noexpand\null##3\noexpand\null
1780 \noexpand\def\noexpand\@gls@xdycheckbackslash{%
1781 \noexpand\@gls@xdycheckbackslash##2\@backslashchar

```

```

1782     \@backslashchar\noexpand\null}%
1783 \noexpand\else
1784     \noexpand\def\noexpand\@gls@xdycheckbackslash{%
1785         \noexpand\@gls@xdycheckbackslash##2\@backslashchar
1786         ##3\noexpand\null}%
1787 \noexpand\fi
1788 \noexpand\fi
1789 \noexpand\@gls@xdycheckbackslash
1790 }%
1791 }

```

Now go ahead and define \@gls@xdycheckbackslash

```

1792 \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`.

```

1793 \ifcsundef{hyperlink}%
1794 {%
1795     \gdef\@glslink#1#2{#2}%
1796 }%
1797 {%
1798     \gdef\@glslink#1#2{\hyperlink{#1}{#2}}%
1799 }

```

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

```

1800 \newlength\gls@tmplen
1801 \ifcsundef{hypertarget}%
1802 {%
1803     \gdef\@glstarget#1#2{#2}%
1804 }%
1805 {%
1806     \gdef\@glstarget#1#2{%
1807         \settoheight{\gls@tmplen}{#2}%
1808         \raisebox{\gls@tmplen}{\hypertarget{#1}{}}#2%
1809     }%
1810 }

```

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

`\glsdisablehyper`

```

1811 \newcommand{\glsdisablehyper}{%
1812 \renewcommand*\@glslink[2]{##2}%
1813 \renewcommand*\@glstarget[2]{##2}}

```

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

`\glsenablehyper`

```
1814 \newcommand{\glsenablehyper}{%
1815 \renewcommand*\@glslink[2]{\hyperlink{##1}{##2}}%
1816 \renewcommand*\@glstarget[2]{%
1817   \settoheight{\gls@tmplen}{##2}%
1818   \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`

```
1819 \newrobustcmd*{\gls}{\@ifstar\@sgls\@gls}
```

Define the starred form:

`\@sgls`

```
1820 \newcommand*{\@sgls}[1][]{\@gls[hyper=false,#1]}
```

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

`\@gls`

```
1821 \newcommand*{\@gls}[2][]{%
1822 \new@ifnextchar{\@gls@{#1}{#2}}{\@gls@{#1}{#2}[]}}
```

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

```
1823 \def\@gls@#1#2[#3]{%
1824 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
1825 \def\@gls@link@opts{#1}%
1826 \def\@gls@link@label{#2}%
```

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

```
1827 \ifglsused{#2}%
1828 {%
1829   \def\@glo@text{%
1830     \csname gls@\@glo@type @display\endcsname
```

```

1831      {\glsentrytext{#2}}{\glsentrydesc{#2}}{\glsentrysymbol{#2}}{#3}}%
1832 }%
1833 {%
1834   \def\@glo@text{%
1835     \csname gls@\@glo@type @displayfirst\endcsname
1836     {\glsentryfirst{#2}}{\glsentrydesc{#2}}{\glsentrysymbol{#2}}{#3}}%
1837 }%

```

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.

```

1838 \ifglsused{#2}{%
1839   \@gls@link[#1]{#2}{\@glo@text}%
1840 }{%
1841   \gls@checkisacronymlist\@glo@type
1842   \ifthenelse{(\boolean{@glsisacronymlist})\AND
1843     \boolean{glsacrfootnote}) \OR \NOT\boolean{glshyperfirst}}{%
1844     \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
1845   }{%
1846     \@gls@link[#1]{#2}{\@glo@text}%
1847   }%
1848 }%

```

Indicate that this entry has now been used

```

1849 \glsunset{#2}}%
1850 }

```

`\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`

```

1851 \newrobustcmd*{\Gls}{\@ifstar\@sGls\@Gls}

```

Define the starred form:

```

1852 \newcommand*{\@sGls}[1][\@Gls[hyper=false,#1]]

```

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

```

1853 \newcommand*{\@Gls}[2][\@Gls@{#1}{#2}]{\@Gls@{#1}{#2}[]}
1854 \new@ifnextchar{\@Gls@{#1}{#2}}{\@Gls@{#1}{#2}[]}

```

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

```

1855 \def\@Gls@#1#2[#3]{%
1856 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
  Save options in \@gls@link@opts and label in \@gls@link@label
1857 \def\@gls@link@opts{#1}%
1858 \def\@gls@link@label{#2}%
1859 \def\glslabel{#2}%

```

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

```

1860 \ifglsused{#2}%
1861 {%
1862   \protected@edef\@glo@text{%
1863     \csname gls@\@glo@type @display\endcsname
1864       {\glsentrytext{#2}}{\glsentrydesc{#2}}%
1865       {\glsentrysymbol{#2}}{#3}}%
1866 }%
1867 {%
1868   \protected@edef\@glo@text{%
1869     \csname gls@\@glo@type @displayfirst\endcsname
1870       {\glsentryfirst{#2}}{\glsentrydesc{#2}}%
1871       {\glsentrysymbol{#2}}{#3}}%
1872 }%

```

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.

```

1873 \ifglsused{#2}{%
1874   \@gls@link[#1]{#2}{%
1875     \expandafter\makefirstuc\expandafter{\@glo@text}}%
1876 }{%
1877   \gls@checkisacronymlist\@glo@type
1878   \ifthenelse{(\boolean{glsisacronymlist}\AND
1879     \boolean{glsacrfootnote})\OR \NOT\boolean{glshyperfirst}}{%
1880     \@gls@link[#1,hyper=false]{#2}{%
1881       \expandafter\makefirstuc\expandafter{\@glo@text}}%
1882   }{%
1883     \@gls@link[#1]{#2}{%
1884       \expandafter\makefirstuc\expandafter{\@glo@text}}%
1885   }%
1886 }%

```

Indicate that this entry has now been used

```

1887 \glsunset{#2}%
1888 }

```

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

\GLS

```

1889 \newrobustcmd*{\GLS}{\@ifstar\@sGLS\@GLS}

```

Define the starred form:

```

1890 \newcommand*{\@sGLS}[1][\@GLS[hyper=false,#1]]

```

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

```

1891 \newcommand*{\@GLS}[2][\@GLS@{#1}{#2}]{\@GLS@{#1}{#2}[]}
1892 \new@ifnextchar[\@GLS@{#1}{#2}]{\@GLS@{#1}{#2}[]}

```

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

```

1893 \def\@GLS@#1#2[#3]{%
1894 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
    Save options in \@gls@link@opts and label in \@gls@link@label
1895 \def\@gls@link@opts{#1}%
1896 \def\@gls@link@label{#2}%
    Determine what the link text should be (this is stored in \@glo@text).
1897 \ifglsused{#2}{\def\@glo@text{%
1898 \csname gls@\@glo@type @display\endcsname
1899 {\glsentrytext{#2}}{\glsentrydesc{#2}}{\glsentrysymbol{#2}}{#3}}}%
1900 \def\@glo@text{%
1901 \csname gls@\@glo@type @displayfirst\endcsname
1902 {\glsentryfirst{#2}}{\glsentrydesc{#2}}{\glsentrysymbol{#2}}{#3}}}%
    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.
1903 \ifglsused{#2}{%
1904   \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
1905 }{%
1906   \gls@checkisacronymlist\@glo@type
1907   \ifthenelse{(\boolean{@glsisacronymlist})\AND
1908     \boolean{glsacrfootnote}) \OR \NOT\boolean{glshyperfirst}}{%
1909     \@gls@link[#1,hyper=false]{#2}{\MakeUppercase{\@glo@text}}%
1910   }{%
1911     \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
1912   }%
1913 }%
    Indicate that this entry has now been used
1914 \glsunset{#2}}%
1915 }

```

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

`\glspl`

```

1916 \newrobustcmd*{\glspl}{\@ifstar\@sglspl\@glspl}

```

Define the starred form:

```

1917 \newcommand*{\@sglspl}[1][\@glspl[hyper=false,#1]]

```

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

```

1918 \newcommand*{\@glspl}[2][\@glspl@{#1}{#2}[]]{%
1919 \new@ifnextchar{\@glspl@{#1}{#2}}{\@glspl@{#1}{#2}[]]}

```

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

```

1920 \def\@glspl@#1#2[#3]{%
1921 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

1922 \def\@gls@link@opts{#1}%
1923 \def\@gls@link@label{#2}%

```

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

```

1924 \ifglsused{#2}%
1925 {%
1926   \def\@glo@text{%
1927     \csname gls@\@glo@type @display\endcsname
1928     {\glsentryplural{#2}}{\glsentrydescplural{#2}}%
1929     {\glsentrysymbolplural{#2}}{#3}}%
1930 }%
1931 {%
1932   \def\@glo@text{%
1933     \csname gls@\@glo@type @displayfirst\endcsname
1934     {\glsentryfirstplural{#2}}{\glsentrydescplural{#2}}%
1935     {\glsentrysymbolplural{#2}}{#3}}%
1936 }%

```

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.

```

1937 \ifglsused{#2}{%
1938   \@gls@link[#1]{#2}{\@glo@text}%
1939 }{%
1940   \gls@checkisacronymlist\@glo@type
1941   \ifthenelse{(\boolean{glsisacronymlist})\AND
1942     \boolean{glsacrfootnote}) \OR \NOT\boolean{glshyperfirst}}{%
1943     \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
1944   }{%
1945     \@gls@link[#1]{#2}{\@glo@text}%
1946   }%
1947 }%

```

Indicate that this entry has now been used

```

1948 \glsunset{#2}%
1949 }

```

`\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`

```

1950 \newrobustcmd*{\Glspl}{\@ifstar\@sGlspl\@Glspl}

```

Define the starred form:

```

1951 \newcommand*{\@sGlspl}[1][\@Glspl[hyper=false,#1]]

```

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


```

1952 \newcommand*{\@Glspl}[2][\%]
1953 \new@ifnextchar[{\@Glspl@{#1}{#2}}{\@Glspl@{#1}{#2}[]}]

```

\@Glspl@ Read in the final optional argument:

```

1954 \def\@Glspl@#1#2[#3]{%
1955 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
    Save options in \@gls@link@opts and label in \@gls@link@label
1956 \def\@gls@link@opts{#1}%
1957 \def\@gls@link@label{#2}%
1958 \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.

```

1959 \ifglsused{#2}%
1960 {%
1961   \protected@edef\@glo@text{%
1962     \csname gls@\@glo@type @display\endcsname
1963     {\glsentryplural{#2}}{\glsentrydescplural{#2}}%
1964     {\glsentrysymbolplural{#2}}{#3}}%
1965 }%
1966 {%
1967   \protected@edef\@glo@text{%
1968     \csname gls@\@glo@type @displayfirst\endcsname
1969     {\glsentryfirstplural{#2}}{\glsentrydescplural{#2}}%
1970     {\glsentrysymbolplural{#2}}{#3}}%
1971 }%

```

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.

```

1972 \ifglsused{#2}{%
1973   \@gls@link[#1]{#2}{%
1974     \expandafter\xmakefirstuc\expandafter{\@glo@text}}%
1975 }{%
1976   \gls@checkisacronymlist\@glo@type
1977   \ifthenelse{(\boolean{glsisacronymlist})\AND
1978     \boolean{glsacrfootnote}) \OR \NOT\boolean{glshyperfirst}}{%
1979     \@gls@link[#1,hyper=false]{#2}{%
1980       \expandafter\xmakefirstuc\expandafter{\@glo@text}}%
1981   }{%
1982     \@gls@link[#1]{#2}{%
1983       \expandafter\xmakefirstuc\expandafter{\@glo@text}}%
1984   }%
1985 }%

```

Indicate that this entry has now been used

```

1986 \glsunset{#2}%
1987 }

```

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

`\GLSp1`

```
1988 \newrobustcmd*{\GLSp1}{\@ifstar\@sGLSp1\@GLSp1}
```

Define the starred form:

```
1989 \newcommand*{\@sGLSp1}[1] [] {\@GLSp1[hyper=false,#1]}
```

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

```
1990 \newcommand*{\@GLSp1}[2] [] {%
```

```
1991 \new@ifnextchar[{\@GLSp1@{#1}{#2}}{\@GLSp1@{#1}{#2} []}]}
```

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

```
1992 \def\@GLSp1@#1#2[#3] {%
```

```
1993 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
1994 \def\@gls@link@opts{#1}%
```

```
1995 \def\@gls@link@label{#2}%
```

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

```
1996 \ifglsused{#2}{\def\@glo@text{%
```

```
1997 \csname gls@\@glo@type @display\endcsname
```

```
1998 {\glsentryplural{#2}}{\glsentrydescplural{#2}}{%
```

```
1999 \glsentrysymbolplural{#2}}{#3}}{%
```

```
2000 \def\@glo@text{%
```

```
2001 \csname gls@\@glo@type @displayfirst\endcsname
```

```
2002 {\glsentryfirstplural{#2}}{\glsentrydescplural{#2}}{%
```

```
2003 \glsentrysymbolplural{#2}}{#3}}}%
```

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.

```
2004 \ifglsused{#2}{%
```

```
2005 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
```

```
2006 }{%
```

```
2007 \gls@checkisacronymlist\@glo@type
```

```
2008 \ifthenelse{(\boolean{glsisacronymlist}\AND
```

```
2009 \boolean{glsacrfootnote}) \OR \NOT\boolean{glshyperfirst}}{%
```

```
2010 \@gls@link[#1,hyper=false]{#2}{\MakeUppercase{\@glo@text}}%
```

```
2011 }{%
```

```
2012 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
```

```
2013 }%
```

```
2014 }%
```

Indicate that this entry has now been used

```
2015 \glsunset{#2}}%
```

```
2016 }
```

`\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:

```
2017 \newrobustcmd*{\glsdisp}{\@ifstar\sglsdisp\glsdisp}
```

Define the starred form:

```
\sgls
```

```
2018 \newcommand*{\sglsdisp}[1] [] {\glsdisp[hyper=false,#1]}
```

Defined the un-starred form.

```
\@glsdisp
```

```
2019 \newcommand*{\@glsdisp}[3] [] {%
```

```
2020   \glsdoifexists{#2}{%
```

```
2021     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2022     \def\@gls@link@opts{#1}%
```

```
2023     \def\@gls@link@label{#2}%
```

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

```
2024     \ifglsused{#2}%
```

```
2025     {%
```

```
2026       \def\@glo@text{%
```

```
2027         \csname gls@\@glo@type @display\endcsname
```

```
2028         {#3}{\glsentrydesc{#2}}{\glsentrysymbol{#2}}{}}%
```

```
2029     }%
```

```
2030     {%
```

```
2031       \def\@glo@text{%
```

```
2032         \csname gls@\@glo@type @displayfirst\endcsname
```

```
2033         {#3}{\glsentrydesc{#2}}{\glsentrysymbol{#2}}{}}%
```

```
2034     }%
```

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.

```
2035     \ifglsused{#2}%
```

```
2036     {%
```

```
2037       \@gls@link[#1]{#2}{\@glo@text}%
```

```
2038     }%
```

```
2039     {%
```

```
2040       \gls@checkisacronymlist\@glo@type
```

```
2041       \ifthenelse{\boolean{glsisacronymlist}\AND
```

```
2042         \boolean{glsacrfootnote}\) \OR \NOT\boolean{glshyperfirst}}%
```

```
2043     {%
```

```
2044       \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
```

```
2045     }%
```

```

2046      {%
2047      \@gls@link[#1]{#2}{\@glo@text}%
2048      }%
2049      }%

```

Indicate that this entry has now been used

```

2050      \glsunset{#2}%
2051      }%
2052 }

```

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

`\gls{text}`

```

2053 \newrobustcmd*{\gls{text}}{\@ifstar\sgls{text}\gls{text}}

```

Define the starred form:

```

2054 \newcommand*{\sgls{text}}[1][]{\@gls{text}[hyper=false,#1]}

```

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

```

2055 \newcommand*{\gls{text}}[2][]{%
2056 \new@ifnextchar[\@gls@text@{#1}{#2}]{\@gls@text@{#1}{#2}[]}}

```

Read in the final optional argument:

```

2057 \def\@gls@text@#1#2[#3]{%
2058 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

```

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

```

2059 \protected@edef\@glo@text{\glsentrytext{#2}}%

```

Call `\@gls@link`

```

2060 \@gls@link[#1]{#2}{\@glo@text#3}%
2061 }%
2062 }

```

`\GLStext` behaves like `\gls{text}` except the text is converted to uppercase.

`\GLStext`

```

2063 \newrobustcmd*{\GLStext}{\@ifstar\sglStext\GLStext}

```

Define the starred form:

```

2064 \newcommand*{\sglStext}[1][]{\@GLStext[hyper=false,#1]}

```

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

```

2065 \newcommand*{\@GLStext}[2][]{%
2066 \new@ifnextchar[\@GLStext@{#1}{#2}]{\@GLStext@{#1}{#2}[]}}

```

Read in the final optional argument:

```

2067 \def\@GLStext@#1#2[#3]{%
2068 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

```

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

```
2069 \protected@edef\@glo@text{\glsentrytext{#2}}%
```

Call \@gls@link

```
2070 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
```

```
2071 }%
```

```
2072 }
```

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

\Glstext

```
2073 \newrobustcmd*{\Glstext}{\@ifstar\@sGlstext\@Glstext}
```

Define the starred form:

```
2074 \newcommand*{\@sGlstext}[1] [] {\@Glstext[hyper=false,#1]}
```

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

```
2075 \newcommand*{\@Glstext}[2] [] {%
```

```
2076 \new@ifnextchar [\@Glstext@{#1}{#2}]{\@Glstext@{#1}{#2} []}}
```

Read in the final optional argument:

```
2077 \def\@Glstext@#1#2[#3] {%
```

```
2078 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2079 \protected@edef\@glo@text{\glsentrytext{#2}}%
```

Call \@gls@link

```
2080 \@gls@link[#1]{#2}{%
```

```
2081 \expandafter\makefirstuc\expandafter{\@glo@text#3}}%
```

```
2082 }%
```

```
2083 }
```

\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

```
2084 \newrobustcmd*{\glsfirst}{\@ifstar\@sglsfirst\@glsfirst}
```

Define the starred form:

```
2085 \newcommand*{\@sglsfirst}[1] [] {\@glsfirst[hyper=false,#1]}
```

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

```
2086 \newcommand*{\@glsfirst}[2] [] {%
```

```
2087 \new@ifnextchar [\@glsfirst@{#1}{#2}]{\@glsfirst@{#1}{#2} []}}
```

Read in the final optional argument:

```
2088 \def\@glsfirst@#1#2[#3] {%
```

```
2089 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2090 \protected@edef\@glo@text{\glsentryfirst{#2}}%
```

Call \@gls@link

```
2091 \@gls@link[#1]{#2}{\@glo@text#3}%
2092 }%
2093 }
```

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

\Glsfirst

```
2094 \newrobustcmd*{\Glsfirst}{\@ifstar\@sGlsfirst\@Glsfirst}
```

Define the starred form:

```
2095 \newcommand*{\@sGlsfirst}[1][\@Glsfirst[hyper=false,#1]]
```

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

```
2096 \newcommand*{\@Glsfirst}[2][\@%
2097 \new@ifnextchar[\@Glsfirst@{#1}{#2}]{\@Glsfirst@{#1}{#2}[]}]
```

Read in the final optional argument:

```
2098 \def\@Glsfirst@#1#2[#3]{%
2099 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2100 \protected@edef\@glo@text{\glsentryfirst{#2}}%
```

Call \@gls@link

```
2101 \@gls@link[#1]{#2}{%
2102   \expandafter\makefirstuc\expandafter{\@glo@text#3}%
2103 }%
2104 }
```

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

\GLSfirst

```
2105 \newrobustcmd*{\GLSfirst}{\@ifstar\@sGLSfirst\@GLSfirst}
```

Define the starred form:

```
2106 \newcommand*{\@sGLSfirst}[1][\@GLSfirst[hyper=false,#1]]
```

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

```
2107 \newcommand*{\@GLSfirst}[2][\@%
2108 \new@ifnextchar[\@GLSfirst@{#1}{#2}]{\@GLSfirst@{#1}{#2}[]}]
```

Read in the final optional argument:

```
2109 \def\@GLSfirst@#1#2[#3]{%
2110 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2111 \protected@edef\@glo@text{\glsentryfirst{#2}}%
```

Call \@gls@link

```

2112 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2113 }%
2114 }

```

\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

```

2115 \newrobustcmd*{\glsplural}{\@ifstar\@sglsplural\@glsplural}

```

Define the starred form:

```

2116 \newcommand*{\@sglsplural}[1][]{\@glsplural[hyper=false,#1]}

```

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

```

2117 \newcommand*{\@glsplural}[2][]{%
2118 \new@ifnextchar[{\@glsplural@{#1}{#2}}{\@glsplural@{#1}{#2}[]}]

```

Read in the final optional argument:

```

2119 \def\@glsplural@#1#2[#3]{%
2120 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2121 \protected@edef\@glo@text{\glsentryplural{#2}}%

```

Call \@gls@link

```

2122 \@gls@link[#1]{#2}{\@glo@text#3}%
2123 }%
2124 }

```

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

\Glsplural

```

2125 \newrobustcmd*{\Glsplural}{\@ifstar\@sGlsplural\@Glsplural}

```

Define the starred form:

```

2126 \newcommand*{\@sGlsplural}[1][]{\@Glsplural[hyper=false,#1]}

```

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

```

2127 \newcommand*{\@Glsplural}[2][]{%
2128 \new@ifnextchar[{\@Glsplural@{#1}{#2}}{\@Glsplural@{#1}{#2}[]}]

```

Read in the final optional argument:

```

2129 \def\@Glsplural@#1#2[#3]{%
2130 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2131 \protected@edef\@glo@text{\glsentryplural{#2}}%

```

Call \@gls@link

```

2132 \@gls@link[#1]{#2}{%
2133   \expandafter\makefirstuc\expandafter{\@glo@text#3}%
2134 }%
2135 }

```

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

\GLSplural

```

2136 \newrobustcmd*{\GLSplural}{\@ifstar\@sGLSplural\@GLSplural}

```

Define the starred form:

```

2137 \newcommand*{\@sGLSplural}[1][\@GLSplural[hyper=false,#1]]{

```

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

```

2138 \newcommand*{\@GLSplural}[2][\@GLSplural@{#1}{#2}[]]{%
2139 \new@ifnextchar[\@GLSplural@{#1}{#2}]{\@GLSplural@{#1}{#2}[]}}

```

Read in the final optional argument:

```

2140 \def\@GLSplural@#1#2[#3]{%
2141 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2142 \protected@edef\@glo@text{\glsentryplural{#2}}%

```

Call \@gls@link

```

2143 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2144 }%
2145 }

```

\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

```

2146 \newrobustcmd*{\glsfirstplural}{\@ifstar\@sglsfirstplural\@glsfirstplural}

```

Define the starred form:

```

2147 \newcommand*{\@sglsfirstplural}[1][\@glsfirstplural[hyper=false,#1]]{

```

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

```

2148 \newcommand*{\@glsfirstplural}[2][\@glsfirstplural@{#1}{#2}[]]{%
2149 \new@ifnextchar[\@glsfirstplural@{#1}{#2}]{\@glsfirstplural@{#1}{#2}[]}}

```

Read in the final optional argument:

```

2150 \def\@glsfirstplural@#1#2[#3]{%
2151 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2152 \protected@edef\@glo@text{\glsentryfirstplural{#2}}%

```


Call \@gls@link

```

2153 \@gls@link[#1]{#2}{\@glo@text#3}%
2154 }%
2155 }

```

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

\Glsfirstplural

```

2156 \newrobustcmd*{\Glsfirstplural}{\@ifstar\@sGlsfirstplural\@Glsfirstplural}

```

Define the starred form:

```

2157 \newcommand*{\@sGlsfirstplural}[1][\@Glsfirstplural[hyper=false,#1]]{

```

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

```

2158 \newcommand*{\@Glsfirstplural}[2][\@Glsfirstplural@#1]{#2}[]{}
2159 \new@ifnextchar[\@Glsfirstplural@#1]{#2}{\@Glsfirstplural@#1}{#2}[]{}

```

Read in the final optional argument:

```

2160 \def\@Glsfirstplural@#1#2[#3]{%
2161 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2162 \protected@edef\@glo@text{\glsentryfirstplural{#2}}%

```

Call \@gls@link

```

2163 \@gls@link[#1]{#2}{%
2164 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2165 }%
2166 }

```

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

\GLSfirstplural

```

2167 \newrobustcmd*{\GLSfirstplural}{\@ifstar\@sGLSfirstplural\@GLSfirstplural}

```

Define the starred form:

```

2168 \newcommand*{\@sGLSfirstplural}[1][\@GLSfirstplural[hyper=false,#1]]{

```

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

```

2169 \newcommand*{\@GLSfirstplural}[2][\@GLSfirstplural@#1]{#2}[]{}
2170 \new@ifnextchar[\@GLSfirstplural@#1]{#2}{\@GLSfirstplural@#1}{#2}[]{}

```

Read in the final optional argument:

```

2171 \def\@GLSfirstplural@#1#2[#3]{%
2172 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2173 \protected@edef\@glo@text{\glsentryfirstplural{#2}}%

```

Call \@gls@link

```
2174 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2175 }%
2176 }
```

\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

```
2177 \newrobustcmd*{\glsname}{\@ifstar\sglsname\@glsname}
```

Define the starred form:

```
2178 \newcommand*{\@sglsname}[1] [] {\@glsname[hyper=false,#1]}
```

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

```
2179 \newcommand*{\@glsname}[2] [] {%
2180 \new@ifnextchar [\@glsname@{#1}{#2}]{\@glsname@{#1}{#2} []}}
```

Read in the final optional argument:

```
2181 \def\@glsname@#1#2[#3] {%
2182 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2183 \protected@edef\@glo@text{\glsentryname{#2}}%
```

Call \@gls@link

```
2184 \@gls@link[#1]{#2}{\@glo@text#3}%
2185 }%
2186 }
```

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

\Glsname

```
2187 \newrobustcmd*{\Glsname}{\@ifstar\sglsname\@Glsname}
```

Define the starred form:

```
2188 \newcommand*{\@sglsname}[1] [] {\@Glsname[hyper=false,#1]}
```

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

```
2189 \newcommand*{\@Glsname}[2] [] {%
2190 \new@ifnextchar [\@Glsname@{#1}{#2}]{\@Glsname@{#1}{#2} []}}
```

Read in the final optional argument:

```
2191 \def\@Glsname@#1#2[#3] {%
2192 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2193 \protected@edef\@glo@text{\glsentryname{#2}}%
```

Call \@gls@link

```

2194 \@gls@link[#1]{#2}{%
2195   \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2196 }%
2197 }

```

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

\GLSname

```

2198 \newrobustcmd*{\GLSname}{\@ifstar\@sGLSname\@GLSname}

```

Define the starred form:

```

2199 \newcommand*{\@sGLSname}[1][\@GLSname[hyper=false,#1]]{

```

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

```

2200 \newcommand*{\@GLSname}[2][\@GLSname[hyper=false,#1]]{
2201 \new@ifnextchar[\@GLSname@{#1}{#2}}{\@GLSname@{#1}{#2}[]}}

```

Read in the final optional argument:

```

2202 \def\@GLSname@#1#2[#3]{%
2203 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2204 \protected@edef\@glo@text{\glsentryname{#2}}%

```

Call \@gls@link

```

2205 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2206 }%
2207 }

```

\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

```

2208 \newrobustcmd*{\glsdesc}{\@ifstar\@sglsdesc\@glsdesc}

```

Define the starred form:

```

2209 \newcommand*{\@sglsdesc}[1][\@glsdesc[hyper=false,#1]]{

```

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

```

2210 \newcommand*{\@glsdesc}[2][\@glsdesc[hyper=false,#1]]{
2211 \new@ifnextchar[\@glsdesc@{#1}{#2}}{\@glsdesc@{#1}{#2}[]}}

```

Read in the final optional argument:

```

2212 \def\@glsdesc@#1#2[#3]{%
2213 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2214 \protected@edef\@glo@text{\glsentrydesc{#2}}%

```

Call \@gls@link
2215 \@gls@link[#1]{#2}{\@glo@text#3}%
2216 }%
2217 }

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

\Glsdesc

2218 \newrobustcmd*{\Glsdesc}{\@ifstar\@sGlsdesc\@Glsdesc}

Define the starred form:

2219 \newcommand*{\@sGlsdesc}[1][]{\@Glsdesc[hyper=false,#1]}

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

2220 \newcommand*{\@Glsdesc}[2][]{%

2221 \new@ifnextchar[\@Glsdesc@{#1}{#2}]{\@Glsdesc@{#1}{#2}[]}}

Read in the final optional argument:

2222 \def\@Glsdesc@#1#2[#3]{%

2223 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

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

2224 \protected@edef\@glo@text{\glsentrydesc{#2}}%

Call \@gls@link

2225 \@gls@link[#1]{#2}{%

2226 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%

2227 }%

2228 }

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

\GLSdesc

2229 \newrobustcmd*{\GLSdesc}{\@ifstar\@sGLSdesc\@GLSdesc}

Define the starred form:

2230 \newcommand*{\@sGLSdesc}[1][]{\@GLSdesc[hyper=false,#1]}

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

2231 \newcommand*{\@GLSdesc}[2][]{%

2232 \new@ifnextchar[\@GLSdesc@{#1}{#2}]{\@GLSdesc@{#1}{#2}[]}}

Read in the final optional argument:

2233 \def\@GLSdesc@#1#2[#3]{%

2234 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

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

2235 \protected@edef\@glo@text{\glsentrydesc{#2}}%

Call \@gls@link

```

2236 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2237 }%
2238 }

```

\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

```

2239 \newrobustcmd*{\glsdescplural}{\@ifstar\sglsdescplural\glsdescplural}

```

Define the starred form:

```

2240 \newcommand*{\@sglsdescplural}[1][\@glsdescplural[hyper=false,#1]]

```

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

```

2241 \newcommand*{\@glsdescplural}[2][\@%
2242 \new@ifnextchar[\@glsdescplural@{#1}{#2}]{\@glsdescplural@{#1}{#2}[]}]

```

Read in the final optional argument:

```

2243 \def\@glsdescplural@#1#2[#3]{%
2244 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

```

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

```

2245 \protected@edef\@glo@text{\glsentrydescplural{#2}}%

```

Call \@gls@link

```

2246 \@gls@link[#1]{#2}{\@glo@text#3}%
2247 }%
2248 }

```

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

\Glsdescplural

```

2249 \newrobustcmd*{\Glsdescplural}{\@ifstar\sglsdescplural\Glsdescplural}

```

Define the starred form:

```

2250 \newcommand*{\@sglsdescplural}[1][\@Glsdescplural[hyper=false,#1]]

```

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

```

2251 \newcommand*{\@Glsdescplural}[2][\@%
2252 \new@ifnextchar[\@Glsdescplural@{#1}{#2}]{\@Glsdescplural@{#1}{#2}[]}]

```

Read in the final optional argument:

```

2253 \def\@Glsdescplural@#1#2[#3]{%
2254 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

```

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

```

2255 \protected@edef\@glo@text{\glsentrydescplural{#2}}%

```

Call `\@gls@link`

```

2256 \@gls@link[#1]{#2}{%
2257   \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2258 }%
2259 }

```

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

`\GLSdescplural`

```

2260 \newrobustcmd*{\GLSdescplural}{\@ifstar\@sGLSdescplural\@GLSdescplural}

```

Define the starred form:

```

2261 \newcommand*{\@sGLSdescplural}[1][\@GLSdescplural[hyper=false,#1]]

```

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

```

2262 \newcommand*{\@GLSdescplural}[2][\@GLSdescplural[hyper=false,#1]]
2263 \new@ifnextchar[\@GLSdescplural@{#1}{#2}]{\@GLSdescplural@{#1}{#2}[]}

```

Read in the final optional argument:

```

2264 \def\@GLSdescplural@#1#2[#3]{%
2265 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2266 \protected@edef\@glo@text{\glsentrydescplural{#2}}%

```

Call `\@gls@link`

```

2267 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2268 }%
2269 }

```

`\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`

```

2270 \newrobustcmd*{\glsymbol}{\@ifstar\@sglsymbol\@glsymbol}

```

Define the starred form:

```

2271 \newcommand*{\@sglsymbol}[1][\@glsymbol[hyper=false,#1]]

```

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

```

2272 \newcommand*{\@glsymbol}[2][\@glsymbol[hyper=false,#1]]
2273 \new@ifnextchar[\@glsymbol@{#1}{#2}]{\@glsymbol@{#1}{#2}[]}

```

Read in the final optional argument:

```

2274 \def\@glsymbol@#1#2[#3]{%
2275 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2276 \protected@edef\@glo@text{\glsentrysymbol{#2}}%

```

Call \@gls@link

```
2277 \@gls@link[#1]{#2}{\@glo@text#3}%  
2278 }%  
2279 }
```

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

\Glsymbol

```
2280 \newrobustcmd*{\Glsymbol}{\@ifstar\@sGlsymbol\@Glsymbol}
```

Define the starred form:

```
2281 \newcommand*{\@sGlsymbol}[1][]{\@Glsymbol[hyper=false,#1]}
```

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

```
2282 \newcommand*{\@Glsymbol}[2][]{%  
2283 \new@ifnextchar[{\@Glsymbol@{#1}{#2}}{\@Glsymbol@{#1}{#2}[]}]}
```

Read in the final optional argument:

```
2284 \def\@Glsymbol@#1#2[#3]{%  
2285 \glsoifexists{#2}{\edef\@glo@type{\glentrytype{#2}}%
```

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

```
2286 \protected@edef\@glo@text{\glentrysymbol{#2}}%
```

Call \@gls@link

```
2287 \@gls@link[#1]{#2}{%  
2288 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%  
2289 }%  
2290 }
```

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

\GLSsymbol

```
2291 \newrobustcmd*{\GLSsymbol}{\@ifstar\@sGLSsymbol\@GLSsymbol}
```

Define the starred form:

```
2292 \newcommand*{\@sGLSsymbol}[1][]{\@GLSsymbol[hyper=false,#1]}
```

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

```
2293 \newcommand*{\@GLSsymbol}[2][]{%  
2294 \new@ifnextchar[{\@GLSsymbol@{#1}{#2}}{\@GLSsymbol@{#1}{#2}[]}]}
```

Read in the final optional argument:

```
2295 \def\@GLSsymbol@#1#2[#3]{%  
2296 \glsoifexists{#2}{\edef\@glo@type{\glentrytype{#2}}%
```

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

```
2297 \protected@edef\@glo@text{\glentrysymbol{#2}}%
```

Call \@gls@link

```
2298 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%  
2299 }%  
2300 }
```

\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

```
2301 \newrobustcmd*{\glssymbolplural}{\@ifstar\sglssymbolplural\@glssymbolplural}
```

Define the starred form:

```
2302 \newcommand*{\@sglssymbolplural}[1][\@glssymbolplural[hyper=false,#1]]
```

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

```
2303 \newcommand*{\@glssymbolplural}[2][\@%  
2304 \new@ifnextchar[\@glssymbolplural@{#1}{#2}]{\@glssymbolplural@{#1}{#2}[]}]
```

Read in the final optional argument:

```
2305 \def\@glssymbolplural@#1#2[#3]{%  
2306 \glsdoifexists{#2}{\edef\@glo@type{\glstrytype{#2}}%
```

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

```
2307 \protected@edef\@glo@text{\glstrytype{#2}}%
```

Call \@gls@link

```
2308 \@gls@link[#1]{#2}{\@glo@text#3}%  
2309 }%  
2310 }
```

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

\Glsymbolplural

```
2311 \newrobustcmd*{\Glsymbolplural}{\@ifstar\sglsymbolplural\@Glsymbolplural}
```

Define the starred form:

```
2312 \newcommand*{\sglsymbolplural}[1][\@Glsymbolplural[hyper=false,#1]]
```

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

```
2313 \newcommand*{\@Glsymbolplural}[2][\@%  
2314 \new@ifnextchar[\@Glsymbolplural@{#1}{#2}]{\@Glsymbolplural@{#1}{#2}[]}]
```

Read in the final optional argument:

```
2315 \def\@Glsymbolplural@#1#2[#3]{%  
2316 \glsdoifexists{#2}{\edef\@glo@type{\glstrytype{#2}}%
```

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

```
2317 \protected@edef\@glo@text{\glstrytype{#2}}%
```


Call \@gls@link

```

2318 \@gls@link[#1]{#2}{%
2319   \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2320 }%
2321 }

```

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

\GLSsymbolplural

```

2322 \newrobustcmd*{\GLSsymbolplural}{\@ifstar\@sGLSsymbolplural\@GLSsymbolplural}

```

Define the starred form:

```

2323 \newcommand*{\@sGLSsymbolplural}[1][\@GLSsymbolplural[hyper=false,#1]]{

```

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

```

2324 \newcommand*{\@GLSsymbolplural}[2][\@GLSsymbolplural@{#1}{#2}[]]{%
2325 \new@ifnextchar[\@GLSsymbolplural@{#1}{#2}]{\@GLSsymbolplural@{#1}{#2}[]]}

```

Read in the final optional argument:

```

2326 \def\@GLSsymbolplural@#1#2[#3]{%
2327 \glsdoifexists{#2}{\edef\@glo@type{\glstrytype{#2}}}%

```

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

```

2328 \protected@edef\@glo@text{\glstrysymbolplural{#2}}%

```

Call \@gls@link

```

2329 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2330 }%
2331 }

```

\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

```

2332 \newrobustcmd*{\glsuseri}{\@ifstar\@sglsuseri\@glsuseri}

```

Define the starred form:

```

2333 \newcommand*{\@sglsuseri}[1][\@glsuseri[hyper=false,#1]]{

```

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

```

2334 \newcommand*{\@glsuseri}[2][\@glsuseri@{#1}{#2}[]]{%
2335 \new@ifnextchar[\@glsuseri@{#1}{#2}]{\@glsuseri@{#1}{#2}[]]}

```

Read in the final optional argument:

```

2336 \def\@glsuseri@#1#2[#3]{%
2337 \glsdoifexists{#2}{\edef\@glo@type{\glstrytype{#2}}}%

```

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

```

2338 \protected@edef\@glo@text{\glstryuseri{#2}}%

```

Call \@gls@link
2339 \@gls@link[#1]{#2}{\@glo@text#3}%
2340 }%
2341 }

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

\Glsuseri

2342 \newrobustcmd*{\Glsuseri}{\@ifstar\@sGlsuseri\@Glsuseri}

Define the starred form:

2343 \newcommand*{\@sGlsuseri}[1][]{\@Glsuseri[hyper=false,#1]}

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

2344 \newcommand*{\@Glsuseri}[2][]{%
2345 \new@ifnextchar[{\@Glsuseri@{#1}{#2}}{\@Glsuseri@{#1}{#2}[]}]}

Read in the final optional argument:

2346 \def\@Glsuseri@#1#2[#3]{%
2347 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

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

2348 \protected@edef\@glo@text{\glsentryuseri{#2}}%

Call \@gls@link

2349 \@gls@link[#1]{#2}{%
2350 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2351 }%
2352 }

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

\GLSuseri

2353 \newrobustcmd*{\GLSuseri}{\@ifstar\@sGLSuseri\@GLSuseri}

Define the starred form:

2354 \newcommand*{\@sGLSuseri}[1][]{\@GLSuseri[hyper=false,#1]}

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

2355 \newcommand*{\@GLSuseri}[2][]{%
2356 \new@ifnextchar[{\@GLSuseri@{#1}{#2}}{\@GLSuseri@{#1}{#2}[]}]}

Read in the final optional argument:

2357 \def\@GLSuseri@#1#2[#3]{%
2358 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

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

2359 \protected@edef\@glo@text{\glsentryuseri{#2}}%

Call \@gls@link

```
2360 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2361 }%
2362 }
```

\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

```
2363 \newrobustcmd*{\glsuserii}{\@ifstar\@sglsuserii\@glsuserii}
```

Define the starred form:

```
2364 \newcommand*{\@sglsuserii}[1][\@glsuserii[hyper=false,#1]]
```

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

```
2365 \newcommand*{\@glsuserii}[2][\@%
2366 \new@ifnextchar[\@glsuserii@{#1}{#2}]{\@glsuserii@{#1}{#2}[]}]
```

Read in the final optional argument:

```
2367 \def\@glsuserii@#1#2[#3]{%
2368 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2369 \protected@edef\@glo@text{\glsentryuserii{#2}}%
```

Call \@gls@link

```
2370 \@gls@link[#1]{#2}{\@glo@text#3}%
2371 }%
2372 }
```

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

\Glsuserii

```
2373 \newrobustcmd*{\Glsuserii}{\@ifstar\@sGlsuserii\@Glsuserii}
```

Define the starred form:

```
2374 \newcommand*{\@sGlsuserii}[1][\@Glsuserii[hyper=false,#1]]
```

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

```
2375 \newcommand*{\@Glsuserii}[2][\@%
2376 \new@ifnextchar[\@Glsuserii@{#1}{#2}]{\@Glsuserii@{#1}{#2}[]}]
```

Read in the final optional argument:

```
2377 \def\@Glsuserii@#1#2[#3]{%
2378 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2379 \protected@edef\@glo@text{\glsentryuserii{#2}}%
```

Call \@gls@link

```

2380 \@gls@link[#1]{#2}{%
2381   \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2382 }%
2383 }

```

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

\GLSuserii

```

2384 \newrobustcmd*{\GLSuserii}{\@ifstar\@sGLSuserii\@GLSuserii}

```

Define the starred form:

```

2385 \newcommand*{\@sGLSuserii}[1][\@GLSuserii[hyper=false,#1]]{

```

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

```

2386 \newcommand*{\@GLSuserii}[2][\@GLSuserii@{#1}{#2}[]]{%
2387 \new@ifnextchar[\@GLSuserii@{#1}{#2}]{\@GLSuserii@{#1}{#2}[]]}

```

Read in the final optional argument:

```

2388 \def\@GLSuserii@#1#2[#3]{%
2389 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2390 \protected@edef\@glo@text{\glsentryuserii{#2}}%

```

Call \@gls@link

```

2391 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2392 }%
2393 }

```

\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

```

2394 \newrobustcmd*{\glsuseriii}{\@ifstar\@sglsuseriii\@glsuseriii}

```

Define the starred form:

```

2395 \newcommand*{\@sglsuseriii}[1][\@glsuseriii[hyper=false,#1]]{

```

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

```

2396 \newcommand*{\@glsuseriii}[2][\@glsuseriii@{#1}{#2}[]]{%
2397 \new@ifnextchar[\@glsuseriii@{#1}{#2}]{\@glsuseriii@{#1}{#2}[]]}

```

Read in the final optional argument:

```

2398 \def\@glsuseriii@#1#2[#3]{%
2399 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2400 \protected@edef\@glo@text{\glsentryuseriii{#2}}%

```

Call \@gls@link
2401 \@gls@link[#1]{#2}{\@glo@text#3}%
2402 }%
2403 }

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

\Glsuseriii

2404 \newrobustcmd*{\Glsuseriii}{\@ifstar\@sGlsuseriii\@Glsuseriii}

Define the starred form:

2405 \newcommand*{\@sGlsuseriii}[1][\@Glsuseriii[hyper=false,#1]]

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

2406 \newcommand*{\@Glsuseriii}[2][\@Glsuseriii[hyper=false,#1]]{%
2407 \new@ifnextchar[\@Glsuseriii@{#1}{#2}]{\@Glsuseriii@{#1}{#2}[]}}

Read in the final optional argument:

2408 \def\@Glsuseriii@#1#2[#3]{%
2409 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

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

2410 \protected@edef\@glo@text{\glsentryuseriii{#2}}%

Call \@gls@link

2411 \@gls@link[#1]{#2}{%
2412 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2413 }%
2414 }

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

\GLSuseriii

2415 \newrobustcmd*{\GLSuseriii}{\@ifstar\@sGLSuseriii\@GLSuseriii}

Define the starred form:

2416 \newcommand*{\@sGLSuseriii}[1][\@GLSuseriii[hyper=false,#1]]

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

2417 \newcommand*{\@GLSuseriii}[2][\@GLSuseriii[hyper=false,#1]]{%
2418 \new@ifnextchar[\@GLSuseriii@{#1}{#2}]{\@GLSuseriii@{#1}{#2}[]}}

Read in the final optional argument:

2419 \def\@GLSuseriii@#1#2[#3]{%
2420 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

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

2421 \protected@edef\@glo@text{\glsentryuseriii{#2}}%

Call \@gls@link

```
2422 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%  
2423 }%  
2424 }
```

\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

```
2425 \newrobustcmd*{\glsuseriv}{\@ifstar\@sglsuseriv\@glsuseriv}
```

Define the starred form:

```
2426 \newcommand*{\@sglsuseriv}[1][\@glsuseriv[hyper=false,#1]]
```

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

```
2427 \newcommand*{\@glsuseriv}[2][\@%  
2428 \new@ifnextchar[\@glsuseriv@{#1}{#2}]{\@glsuseriv@{#1}{#2}[]}}
```

Read in the final optional argument:

```
2429 \def\@glsuseriv@#1#2[#3]{%  
2430 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2431 \protected@edef\@glo@text{\glsentryuseriv{#2}}%
```

Call \@gls@link

```
2432 \@gls@link[#1]{#2}{\@glo@text#3}%  
2433 }%  
2434 }
```

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

\Glsuseriv

```
2435 \newrobustcmd*{\Glsuseriv}{\@ifstar\@sGlsuseriv\@Glsuseriv}
```

Define the starred form:

```
2436 \newcommand*{\@sGlsuseriv}[1][\@Glsuseriv[hyper=false,#1]]
```

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

```
2437 \newcommand*{\@Glsuseriv}[2][\@%  
2438 \new@ifnextchar[\@Glsuseriv@{#1}{#2}]{\@Glsuseriv@{#1}{#2}[]}}
```

Read in the final optional argument:

```
2439 \def\@Glsuseriv@#1#2[#3]{%  
2440 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2441 \protected@edef\@glo@text{\glsentryuseriv{#2}}%
```

Call \@gls@link

```

2442 \@gls@link[#1]{#2}{%
2443   \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2444 }%
2445 }

```

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

\GLSuseriv

```

2446 \newrobustcmd*{\GLSuseriv}{\@ifstar\@sGLSuseriv\@GLSuseriv}

```

Define the starred form:

```

2447 \newcommand*{\@sGLSuseriv}[1][\@GLSuseriv[hyper=false,#1]]{

```

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

```

2448 \newcommand*{\@GLSuseriv}[2][\@GLSuseriv@{#1}{#2}[]]{%
2449 \new@ifnextchar[\@GLSuseriv@{#1}{#2}]{\@GLSuseriv@{#1}{#2}[]}}

```

Read in the final optional argument:

```

2450 \def\@GLSuseriv@#1#2[#3]{%
2451 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2452 \protected@edef\@glo@text{\glsentryuseriv{#2}}%

```

Call \@gls@link

```

2453 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2454 }%
2455 }

```

\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

```

2456 \newrobustcmd*{\glsuserv}{\@ifstar\@sglsuserv\@glsuserv}

```

Define the starred form:

```

2457 \newcommand*{\@sglsuserv}[1][\@glsuserv[hyper=false,#1]]{

```

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

```

2458 \newcommand*{\@glsuserv}[2][\@glsuserv@{#1}{#2}[]]{%
2459 \new@ifnextchar[\@glsuserv@{#1}{#2}]{\@glsuserv@{#1}{#2}[]}}

```

Read in the final optional argument:

```

2460 \def\@glsuserv@#1#2[#3]{%
2461 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2462 \protected@edef\@glo@text{\glsentryuseriv{#2}}%

```

Call \@gls@link
2463 \@gls@link[#1]{#2}{\@glo@text#3}%
2464 }%
2465 }

\Glsuserv behaves like \glsuserv except that the first letter is converted to uppercase.

\Glsuserv

2466 \newrobustcmd*{\Glsuserv}{\@ifstar\@sGlsuserv\@Glsuserv}

Define the starred form:

2467 \newcommand*{\@sGlsuserv}[1][]{\@Glsuserv[hyper=false,#1]}

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

2468 \newcommand*{\@Glsuserv}[2][]{%
2469 \new@ifnextchar[{\@Glsuserv@{#1}{#2}}{\@Glsuserv@{#1}{#2}[]}]}

Read in the final optional argument:

2470 \def\@Glsuserv@#1#2[#3]{%
2471 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

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

2472 \protected@edef\@glo@text{\glsentryuserv{#2}}%

Call \@gls@link

2473 \@gls@link[#1]{#2}{%
2474 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2475 }%
2476 }

\GLSuserv behaves like \glsuserv except that the link text is converted to uppercase.

\GLSuserv

2477 \newrobustcmd*{\GLSuserv}{\@ifstar\@sGLSuserv\@GLSuserv}

Define the starred form:

2478 \newcommand*{\@sGLSuserv}[1][]{\@GLSuserv[hyper=false,#1]}

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

2479 \newcommand*{\@GLSuserv}[2][]{%
2480 \new@ifnextchar[{\@GLSuserv@{#1}{#2}}{\@GLSuserv@{#1}{#2}[]}]}

Read in the final optional argument:

2481 \def\@GLSuserv@#1#2[#3]{%
2482 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

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

2483 \protected@edef\@glo@text{\glsentryuserv{#2}}%

Call \@gls@link
2484 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2485 }%
2486 }

\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

2487 \newrobustcmd*{\glsuservi}{\@ifstar\@sglsuservi\@glsuservi}

Define the starred form:

2488 \newcommand*{\@sglsuservi}[1][\@glsuservi[hyper=false,#1]]

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

2489 \newcommand*{\@glsuservi}[2][\@glsuservi@#1]{#2}%
2490 \new@ifnextchar[\@glsuservi@{\@glsuservi@#1}{#2}[]}]

Read in the final optional argument:

2491 \def\@glsuservi@#1#2[#3]{%
2492 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

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

2493 \protected@edef\@glo@text{\glsentryuservi{#2}}%

Call \@gls@link

2494 \@gls@link[#1]{#2}{\@glo@text#3}}%
2495 }%
2496 }

\Glsuservi behaves like \glsuservi except that the first letter is converted to uppercase.

\Glsuservi

2497 \newrobustcmd*{\Glsuservi}{\@ifstar\@sGlsuservi\@Glsuservi}

Define the starred form:

2498 \newcommand*{\@sGlsuservi}[1][\@Glsuservi[hyper=false,#1]]

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

2499 \newcommand*{\@Glsuservi}[2][\@Glsuservi@#1]{#2}%
2500 \new@ifnextchar[\@Glsuservi@{\@Glsuservi@#1}{#2}[]}]

Read in the final optional argument:

2501 \def\@Glsuservi@#1#2[#3]{%
2502 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

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

2503 \protected@edef\@glo@text{\glsentryuservi{#2}}%

Call \@gls@link
2504 \@gls@link[#1]{#2}{%
2505 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2506 }%
2507 }
\GLSuservi behaves like \glsuservi except that the link text is converted to uppercase.

\GLSuservi

2508 \newrobustcmd*{\GLSuservi}{\@ifstar\@sGLSuservi\@GLSuservi}

Define the starred form:

2509 \newcommand*{\@sGLSuservi}[1] [] {\@GLSuservi[hyper=false,#1]}

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

2510 \newcommand*{\@GLSuservi}[2] [] {%

2511 \new@ifnextchar [\@GLSuservi@{#1}{#2}]{\@GLSuservi@{#1}{#2} []}}

Read in the final optional argument:

2512 \def\@GLSuservi@#1#2[#3] {%

2513 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

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

2514 \protected@edef\@glo@text{\glsentryuservi{#2}}%

Call \@gls@link

2515 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%

2516 }%

2517 }

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

\acrshort

2518 \newrobustcmd*{\acrshort}{\@ifstar\s@acrshort\@ns@acrshort}

Define the starred form:

2519 \newcommand*{\s@acrshort}[2] [] {%

2520 \new@ifnextchar [\@acrshort{hyper=false,#1}{#2}}%

2521 {\@acrshort{hyper=false,#1}{#2} []}%

2522 }

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

2523 \newcommand*{\@ns@acrshort}[2] [] {%

2524 \new@ifnextchar [\@acrshort{#1}{#2}]{\@acrshort{#1}{#2} []}%

2525 }

Read in the final optional argument:

2526 \def\@acrshort#1#2[#3] {%

2527 \glsdoifexists{#2}%

2528 {%

2529 \edef\@glo@type{\glsentrytype{#2}}%

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

```
2530 \protected@edef\@glo@text{\glentryshort{#2}}%  
    Call \@gls@link  
2531 \@gls@link[#1]{#2}{\acronymfont{\@glo@text}#3}%  
2532 }%  
2533 }
```

\Acrshort

```
2534 \newrobustcmd*{\Acrshort}{\@ifstar\s@Acrshort\ns@Acrshort}
```

Define the starred form:

```
2535 \newcommand*{\s@Acrshort}[2][]{%  
2536 \new@ifnextchar[{\@Acrshort{hyper=false,#1}{#2}}%  
2537 {\@Acrshort{hyper=false,#1}{#2}[]}%  
2538 }
```

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

```
2539 \newcommand*{\ns@Acrshort}[2][]{%  
2540 \new@ifnextchar[{\@Acrshort{#1}{#2}}{\@Acrshort{#1}{#2}[]}%  
2541 }
```

Read in the final optional argument:

```
2542 \def\@Acrshort#1#2[#3]{%  
2543 \glsoifexists{#2}%  
2544 {%  
2545 \edef\@glo@type{\glentrytype{#2}}%
```

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

```
2546 \protected@edef\@glo@text{\glentryshort{#2}}%  
    Call \@gls@link  
2547 \@gls@link[#1]{#2}%  
2548 {%  
2549 \acronymfont{\expandafter\makefirstuc\expandafter{\@glo@text}}#3%  
2550 }%  
2551 }%  
2552 }
```

\ACRshort

```
2553 \newrobustcmd*{\ACRshort}{\@ifstar\s@ACRshort\ns@ACRshort}
```

Define the starred form:

```
2554 \newcommand*{\s@ACRshort}[2][]{%  
2555 \new@ifnextchar[{\@ACRshort{hyper=false,#1}{#2}}%  
2556 {\@ACRshort{hyper=false,#1}{#2}[]}%  
2557 }
```

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

```
2558 \newcommand*{\ns@ACRshort}[2] [] {%
2559   \new@ifnextchar[{\@ACRshort{#1}{#2}}{\@ACRshort{#1}{#2} []}%
2560 }
```

Read in the final optional argument:

```
2561 \def\@ACRshort#1#2[#3] {%
2562   \glsdoifexists{#2}%
2563   {%
2564     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2565   \protected@edef\@glo@text{\glsentryshort{#2}}%
```

Call \@gls@link

```
2566   \@gls@link[#1]{#2}{\acronymfont{\MakeUppercase{\@glo@text#3}}}%
2567   }%
2568 }
```

Short plural:

\acrshortpl

```
2569 \newrobustcmd*{\acrshortpl}{\@ifstar\s@acrshortpl\ns@acrshortpl}
```

Define the starred form:

```
2570 \newcommand*{\s@acrshortpl}[2] [] {%
2571   \new@ifnextchar[{\@acrshortpl{hyper=false,#1}{#2}}%
2572   {\@acrshortpl{hyper=false,#1}{#2} []}%
2573 }
```

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

```
2574 \newcommand*{\ns@acrshortpl}[2] [] {%
2575   \new@ifnextchar[{\@acrshortpl{#1}{#2}}{\@acrshortpl{#1}{#2} []}%
2576 }
```

Read in the final optional argument:

```
2577 \def\@acrshortpl#1#2[#3] {%
2578   \glsdoifexists{#2}%
2579   {%
2580     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2581   \protected@edef\@glo@text{\glsentryshortpl{#2}}%
```

Call \@gls@link

```
2582   \@gls@link[#1]{#2}{\acronymfont{\@glo@text#3}}%
2583   }%
2584 }
```

\Acrshortpl

```
2585 \newrobustcmd*{\Acrshortpl}{\@ifstar\s@Acrshortpl\ns@Acrshortpl}
```

Define the starred form:

```
2586 \newcommand*{\s@Acrshortpl}[2] [] {%
2587   \new@ifnextchar[{\@Acrshortpl{hyper=false,#1}{#2}}}%
2588   {\@Acrshortpl{hyper=false,#1}{#2} []}%
2589 }
```

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

```
2590 \newcommand*{\ns@Acrshortpl}[2] [] {%
2591   \new@ifnextchar[{\@Acrshortpl{#1}{#2}}{\@Acrshortpl{#1}{#2} []}%
2592 }
```

Read in the final optional argument:

```
2593 \def\@Acrshortpl#1#2[#3] {%
2594   \glsdoifexists{#2}%
2595   {%
2596     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2597   \protected@edef\@glo@text{\glsentryshortpl{#2}}%
```

Call \@gls@link

```
2598   \@gls@link[#1]{#2}%
2599   {%
2600     \acronymfont{\expandafter\makefirstuc\expandafter{\@glo@text}}#3%
2601   }%
2602 }%
2603 }
```

\ACRshortpl

```
2604 \newrobustcmd*{\ACRshortpl}{\@ifstar\s@ACRshortpl\ns@ACRshortpl}
```

Define the starred form:

```
2605 \newcommand*{\s@ACRshortpl}[2] [] {%
2606   \new@ifnextchar[{\@ACRshortpl{hyper=false,#1}{#2}}}%
2607   {\@ACRshortpl{hyper=false,#1}{#2} []}%
2608 }
```

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

```
2609 \newcommand*{\ns@ACRshortpl}[2] [] {%
2610   \new@ifnextchar[{\@ACRshortpl{#1}{#2}}{\@ACRshortpl{#1}{#2} []}%
2611 }
```

Read in the final optional argument:

```
2612 \def\@ACRshortpl#1#2[#3] {%
2613   \glsdoifexists{#2}%
2614   {%
2615     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2616   \protected@edef\@glo@text{\glsentryshortpl{#2}}%
```

Call \@gls@link

```

2617   \@gls@link[#1]{#2}{\acronymfont{\MakeUppercase{\@glo@text#3}}}%
2618   }%
2619 }

```

\acrlong

```

2620 \newrobustcmd*{\acrlong}{\@ifstar\s@acrlong\ns@acrlong}

```

Define the starred form:

```

2621 \newcommand*{\s@acrlong}[2][]{%
2622   \new@ifnextchar[{\@acrlong{hyper=false,#1}{#2}}%
2623     {\@acrlong{hyper=false,#1}{#2}[]}%
2624 }

```

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

```

2625 \newcommand*{\ns@acrlong}[2][]{%
2626   \new@ifnextchar[{\@acrlong{#1}{#2}}{\@acrlong{#1}{#2}[]}%
2627 }

```

Read in the final optional argument:

```

2628 \def\@acrlong#1#2[#3]{%
2629   \glsdoifexists{#2}%
2630   {%
2631     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2632   \protected@edef\@glo@text{\glsentrylong{#2}}%

```

Call \@gls@link

```

2633   \@gls@link[#1]{#2}{\@glo@text#3}%
2634   }%
2635 }

```

\Acrlong

```

2636 \newrobustcmd*{\Acrlong}{\@ifstar\s@Acrlong\ns@Acrlong}

```

Define the starred form:

```

2637 \newcommand*{\s@Acrlong}[2][]{%
2638   \new@ifnextchar[{\@Acrlong{hyper=false,#1}{#2}}%
2639     {\@Acrlong{hyper=false,#1}{#2}[]}%
2640 }

```

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

```

2641 \newcommand*{\ns@Acrlong}[2][]{%
2642   \new@ifnextchar[{\@Acrlong{#1}{#2}}{\@Acrlong{#1}{#2}[]}%
2643 }

```

Read in the final optional argument:

```
2644 \def\@Acrlong#1#2[#3]{%
2645   \glsdoifexists{#2}%
2646   {%
2647     \edef\@glo@type{\glstrytype{#2}}%
2648     \protected@edef\@glo@text{\glstrylong{#2}}%
2649     \Call\@gls@link
2650     {%
2651       \expandafter\makefirstuc\expandafter{\@glo@text}#3%
2652     }%
2653   }%
2654 }
```

\ACRlong

```
2655 \newrobustcmd*{\ACRlong}{\@ifstar\s@ACRlong\@ns@ACRlong}
```

Define the starred form:

```
2656 \newcommand*{\s@ACRlong}[2][]{%
2657   \new@ifnextchar[{\@ACRlong{hyper=false,#1}{#2}}%
2658   {\@ACRlong{hyper=false,#1}{#2} []}%
2659 }
```

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

```
2660 \newcommand*{\ns@ACRlong}[2][]{%
2661   \new@ifnextchar[{\@ACRlong{#1}{#2}}{\@ACRlong{#1}{#2} []}%
2662 }
```

Read in the final optional argument:

```
2663 \def\@ACRlong#1#2[#3]{%
2664   \glsdoifexists{#2}%
2665   {%
2666     \edef\@glo@type{\glstrytype{#2}}%
2667     \protected@edef\@glo@text{\glstrylong{#2}}%
2668     \Call\@gls@link
2669     {%
2670       \MakeUppercase{\@glo@text#3}%
2671     }%
2672   }%
2673 }
```

Short plural:

\acrlongpl

```
2671 \newrobustcmd*{\acrlongpl}{\@ifstar\s@acrlongpl\@ns@acrlongpl}
```

Define the starred form:

```
2672 \newcommand*{\s@acrlongpl}[2] [] {%
2673   \new@ifnextchar[{\@acrlongpl{hyper=false,#1}{#2}}%
2674     {\@acrlongpl{hyper=false,#1}{#2} []}%
2675 }
```

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

```
2676 \newcommand*{\ns@acrlongpl}[2] [] {%
2677   \new@ifnextchar[{\@acrlongpl{#1}{#2}}{\@acrlongpl{#1}{#2} []}%
2678 }
```

Read in the final optional argument:

```
2679 \def\@acrlongpl#1#2[#3] {%
2680   \glsdoifexists{#2}%
2681   {%
2682     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2683   \protected@edef\@glo@text{\glsentrylongpl{#2}}%
```

Call \@gls@link

```
2684   \@gls@link[#1]{#2}{\@glo@text#3}%
2685 }%
2686 }
```

\Acrlongpl

```
2687 \newrobustcmd*{\Acrlongpl}{\@ifstar\s@Acrlongpl\ns@Acrlongpl}
```

Define the starred form:

```
2688 \newcommand*{\s@Acrlongpl}[2] [] {%
2689   \new@ifnextchar[{\@Acrlongpl{hyper=false#1}{#2}}%
2690     {\@Acrlongpl{hyper=false,#1}{#2} []}%
2691 }
```

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

```
2692 \newcommand*{\ns@Acrlongpl}[2] [] {%
2693   \new@ifnextchar[{\@Acrlongpl{#1}{#2}}{\@Acrlongpl{#1}{#2} []}%
2694 }
```

Read in the final optional argument:

```
2695 \def\@Acrlongpl#1#2[#3] {%
2696   \glsdoifexists{#2}%
2697   {%
2698     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2699   \protected@edef\@glo@text{\glsentrylongpl{#2}}%
```



```

Call \@gls@link
2700   \@gls@link[#1]{#2}%
2701   {%
2702     \expandafter\makefirstuc\expandafter{\@glo@text}#3%
2703   }%
2704 }%
2705 }

```

`\ACRlongpl`

```

2706 \newrobustcmd*{\ACRlongpl}{\@ifstar\s@ACRlongpl\ns@ACRlongpl}

```

Define the starred form:

```

2707 \newcommand*{\s@ACRlongpl}[2][ ]{%
2708   \new@ifnextchar[{\@ACRlongpl{hyper=false,#1}{#2}}%
2709     {\@ACRlongpl{hyper=false,#1}{#2}[ ]}%
2710 }

```

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

```

2711 \newcommand*{\ns@ACRlongpl}[2][ ]{%
2712   \new@ifnextchar[{\@ACRlongpl{#1}{#2}}{\@ACRlongpl{#1}{#2}[ ]}%
2713 }

```

Read in the final optional argument:

```

2714 \def\@ACRlongpl#1#2[#3]{%
2715   \glsdoifexists{#2}%
2716   {%
2717     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2718   \protected@edef\@glo@text{\glsentrylongpl{#2}}%

```

Call \@gls@link

```

2719   \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2720 }%
2721 }

```

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`

```

2722 \newcommand*{\glsentryname}[1]{\csname glo@#1@name\endcsname}

```

`\Glsentryname`

```
2723 \newcommand*{\Glsentryname}[1]{%
2724 \protected@edef\@glo@text{\csname glo@#1@name\endcsname}%
2725 \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`

```
2726 \newcommand*{\glsentrydesc}[1]{\csname glo@#1@desc\endcsname}
```

`\Glsentrydesc`

```
2727 \newcommand*{\Glsentrydesc}[1]{%
2728 \protected@edef\@glo@text{\csname glo@#1@desc\endcsname}%
2729 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Plural form:

`\glsentrydescplural`

```
2730 \newcommand*{\glsentrydescplural}[1]{%
2731 \csname glo@#1@descplural\endcsname}
```

`\Glsentrydescplural`

```
2732 \newcommand*{\Glsentrydescplural}[1]{%
2733 \protected@edef\@glo@text{\csname glo@#1@descplural\endcsname}%
2734 \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`

```
2735 \newcommand*{\glsentrytext}[1]{\csname glo@#1@text\endcsname}
```

`\Glsentrytext`

```
2736 \newcommand*{\Glsentrytext}[1]{%
2737 \protected@edef\@glo@text{\csname glo@#1@text\endcsname}%
2738 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Get the plural form:

`\glsentryplural`

```
2739 \newcommand*{\glsentryplural}[1]{\csname glo@#1@plural\endcsname}
```

`\Glsentryplural`

```
2740 \newcommand*{\Glsentryplural}[1]{%
2741 \protected@edef\@glo@text{\csname glo@#1@plural\endcsname}%
2742 \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`

```
2743 \newcommand*{\glsentrysymbol}[1]{\csname glo@#1@symbol\endcsname}
```

`\Glsentrysymbol`

```
2744 \newcommand*{\Glsentrysymbol}[1]{%
2745 \protected@edef\@glo@text{\csname glo@#1@symbol\endcsname}%
2746 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Plural form:

`\glsentrysymbolplural`

```
2747 \newcommand*{\glsentrysymbolplural}[1]{%
2748 \csname glo@#1@symbolplural\endcsname}
```

`\Glsentrysymbolplural`

```
2749 \newcommand*{\Glsentrysymbolplural}[1]{%
2750 \protected@edef\@glo@text{\csname glo@#1@symbolplural\endcsname}%
2751 \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`

```
2752 \newcommand*{\glsentryfirst}[1]{\csname glo@#1@first\endcsname}
```

`\Glsentryfirst`

```
2753 \newcommand*{\Glsentryfirst}[1]{%
2754 \protected@edef\@glo@text{\csname glo@#1@first\endcsname}%
2755 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Get the plural form (as specified by the `firstplural` key when the entry was defined).

`\glsentryfirstplural`

```
2756 \newcommand*{\glsentryfirstplural}[1]{%
2757 \csname glo@#1@firstpl\endcsname}
```

`\Glsentryfirstplural`

```
2758 \newcommand*{\Glsentryfirstplural}[1]{%
2759 \protected@edef\@glo@text{\csname glo@#1@firstpl\endcsname}%
2760 \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)

`\glsentrytype`

2761 `\newcommand*{\glsentrytype}[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.

`\glsentrysort`

2762 `\newcommand*{\glsentrysort}[1]{\csname glo@#1@sort\endcsname}`

`\glsentryuseri` Get the first user key (as specified by the user1 when the entry was defined). The argument is the label associated with the entry.

2763 `\newcommand*{\glsentryuseri}[1]{\csname glo@#1@useri\endcsname}`

`\Glsentryuseri`

2764 `\newcommand*{\Glsentryuseri}[1]{%`
2765 `\protected@edef\@glo@text{\csname glo@#1@useri\endcsname}%`
2766 `\expandafter\makefirstuc\expandafter{\@glo@text}}`

`\glsentryuserii` Get the second user key (as specified by the user2 when the entry was defined). The argument is the label associated with the entry.

2767 `\newcommand*{\glsentryuserii}[1]{\csname glo@#1@userii\endcsname}`

`\Glsentryuserii`

2768 `\newcommand*{\Glsentryuserii}[1]{%`
2769 `\protected@edef\@glo@text{\csname glo@#1@userii\endcsname}%`
2770 `\expandafter\makefirstuc\expandafter{\@glo@text}}`

`\glsentryuseriii` Get the third user key (as specified by the user3 when the entry was defined). The argument is the label associated with the entry.

2771 `\newcommand*{\glsentryuseriii}[1]{\csname glo@#1@useriii\endcsname}`

`\Glsentryuseriii`

2772 `\newcommand*{\Glsentryuseriii}[1]{%`
2773 `\protected@edef\@glo@text{\csname glo@#1@useriii\endcsname}%`
2774 `\expandafter\makefirstuc\expandafter{\@glo@text}}`

`\glsentryuseriv` Get the fourth user key (as specified by the user4 when the entry was defined). The argument is the label associated with the entry.

2775 `\newcommand*{\glsentryuseriv}[1]{\csname glo@#1@useriv\endcsname}`

`\Glsentryuseriv`

2776 `\newcommand*{\Glsentryuseriv}[1]{%`
2777 `\protected@edef\@glo@text{\csname glo@#1@useriv\endcsname}%`
2778 `\expandafter\makefirstuc\expandafter{\@glo@text}}`

`\glsentryuserv` Get the fifth user key (as specified by the user5 when the entry was defined). The argument is the label associated with the entry.

2779 `\newcommand*{\glsentryuserv}[1]{\csname glo@#1@userv\endcsname}`

```

\Glsentryuserv
2780 \newcommand*{\Glsentryuserv}[1]{%
2781 \protected@edef\@glo@text{\csname glo@#1@userv\endcsname}%
2782 \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.
2783 \newcommand*{\glsentryuservi}[1]{\csname glo@#1@uservi\endcsname}

\Glsentryuservi
2784 \newcommand*{\Glsentryuservi}[1]{%
2785 \protected@edef\@glo@text{\csname glo@#1@uservi\endcsname}%
2786 \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.
2787 \newcommand*{\glsentryshort}[1]{\csname glo@#1@short\endcsname}

\Glsentryshort
2788 \newcommand*{\Glsentryshort}[1]{%
2789 \protected@edef\@glo@text{\csname glo@#1@short\endcsname}%
2790 \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.
2791 \newcommand*{\glsentryshortpl}[1]{\csname glo@#1@shortpl\endcsname}

\Glsentryshortpl
2792 \newcommand*{\Glsentryshortpl}[1]{%
2793 \protected@edef\@glo@text{\csname glo@#1@shortpl\endcsname}%
2794 \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.
2795 \newcommand*{\glsentrylong}[1]{\csname glo@#1@long\endcsname}

\Glsentrylong
2796 \newcommand*{\Glsentrylong}[1]{%
2797 \protected@edef\@glo@text{\csname glo@#1@long\endcsname}%
2798 \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.
2799 \newcommand*{\glsentrylongpl}[1]{\csname glo@#1@longpl\endcsname}

```

`\Glsentrylongpl`

```
2800 \newcommand*{\Glsentrylongpl}[1]{%
2801 \protected@edef\@glo@text{\csname glo@#1@longpl\endcsname}%
2802 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Short cut macros to access full form:

`\glsentryfull`

```
2803 \newcommand*{\glsentryfull}[1]{%
2804 \glsentrylong{#1}\space\glsentryshort{#1})%
2805 }
```

`\Glsentryfull`

```
2806 \newcommand*{\Glsentryfull}[1]{%
2807 \Glsentrylong{#1}\space\glsentryshort{#1})%
2808 }
```

`\glsentryfullpl`

```
2809 \newcommand*{\glsentryfullpl}[1]{%
2810 \glsentrylongpl{#1}\space\glsentryshortpl{#1})%
2811 }
```

`\Glsentryfullpl`

```
2812 \newcommand*{\Glsentryfullpl}[1]{%
2813 \Glsentrylongpl{#1}\space\glsentryshortpl{#1})%
2814 }
```

`\glsentrynumberlist` Displays the number list as is.

```
2815 \newcommand*{\glsentrynumberlist}[1]{%
2816 \glsdoifexists{#1}%
2817 {%
2818 \csname glo@#1@numberlist\endcsname
2819 }%
2820 }
```

`\glsdisplaynumberlist` Formats the number list for the given entry label. Doesn't work with hyperref.

```
2821 \@ifpackageloaded{hyperref}
2822 {%
2823 \newcommand*{\glsdisplaynumberlist}[1]{%
2824 \GlossariesWarning
2825 {%
2826 \string\glsdisplaynumberlist\space
2827 doesn't work with hyperref.^^JUsing
2828 \string\glsentrynumberlist\space instead%
2829 }%
2830 \glsentrynumberlist{#1}%
2831 }%
2832 }
```

```

2833 {%
2834   \newcommand*{\glsdisplaynumberlist}[1]{%
2835     \glsdoifexists{#1}%
2836     {%
2837       \bgroup
2838         \def\@glo@label{#1}%
2839         \let\@org@glnumberformat\glnumberformat
2840         \def\glnumberformat##1{##1}%
2841         \protected@edef\the@numberlist{\csname glo@\@glo@label @numberlist\endcsname}%
2842         \def\@gls@numlist@sep{}%
2843         \def\@gls@numlist@nextsep{}%
2844         \def\@gls@numlist@lastsep{}%
2845         \def\@gls@thislist{}%
2846         \def\@gls@donext@def{}%
2847         \renewcommand\do[1]{%
2848           \protected@edef\@gls@thislist{%
2849             \@gls@thislist
2850             \noexpand\@gls@numlist@sep
2851             ##1%
2852           }%
2853           \let\@gls@numlist@sep\@gls@numlist@nextsep
2854           \def\@gls@numlist@nextsep{\glnumlistsep}%
2855           \@gls@donext@def
2856           \def\@gls@donext@def{%
2857             \def\@gls@numlist@lastsep{\glnumlistlastsep}%
2858           }%
2859         }%
2860         \expandafter \glnumlistparser \expandafter{\the@numberlist}%
2861         \let\@gls@numlist@sep\@gls@numlist@lastsep
2862         \@gls@thislist
2863       \egroup
2864     }%
2865   }
2866 }

```

`\glnumlistsep`

```
2867 \newcommand*{\glnumlistsep}{, }
```

`\glnumlistlastsep`

```
2868 \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.

```

2869 \newcommand*{\glshyperlink}[2][\glentrytext{\@glo@label}]{%
2870 \def\@glo@label{#2}%
2871 \@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`:

```
2872 \define@key{glossadd}{counter}{\def\@gls@counter{#1}}
```

```
2873 \define@key{glossadd}{format}{\def\@glsnumberformat{#1}}
```

This key is only used by `\glsaddall`:

```
2874 \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`

```
2875 \newrobustcmd*{\glsadd}[2][]{%
```

```
2876   \glsdoifexists{#2}%
```

```
2877   {%
```

```
2878     \def\@glsnumberformat{glsnumberformat}%
```

```
2879     \edef\@gls@counter{\csname glo@#2@counter\endcsname}%
```

```
2880     \setkeys{glossadd}{#1}%
```

Store the entry's counter in `\theglsentrycounter`

```
2881     \@gls@saveentrycounter
```

```
2882     \@do@wrglossary{#2}%
```

```
2883   }%
```

```
2884 }
```

`\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`

```
2885 \newrobustcmd*{\glsaddall}[1][]{%
```

```
2886   \edef\@glo@type{\@glo@types}%
```

```
2887   \setkeys{glossadd}{#1}%
```

```
2888   \forallglsentries[\@glo@type]{\@glo@entry}{%
```

```
2889     \glsadd[#1]{\@glo@entry}}%
```

```
2890 }
```

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.

```
2891 \edef\glsopenbrace{\expandafter\@gobble\string\{}
```

`\glsclosebrace` Define \glsclosebrace to make it easier to write an opening brace to a file.

```
2892 \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.

```
2893 \edef\glsquote#1{\string"#1\string"}
```

`\@glsfirstletter` Define the first letter to come after the digits 0,...,9. Only required for xindy.

```
2894 \ifglsxindy
```

```
2895   \newcommand*{\@glsfirstletter}{A}
```

```
2896 \fi
```

`stLetterAfterDigits` Sets the first letter to come after the digits 0,...,9.

```
2897 \ifglsxindy
```

```
2898   \newcommand*{\GlsSetXdyFirstLetterAfterDigits}[1]{%
```

```
2899     \renewcommand*{\@glsfirstletter}{#1}}
```

```
2900 \else
```

```
2901   \newcommand*{\GlsSetXdyFirstLetterAfterDigits}[1]{%
```

```
2902     \glsnoxindywarning\GlsSetXdyFirstLetterAfterDigits}
```

```
2903 \fi
```

`\@glsminrange` Define the minimum number of successive location references to merge into a range.

```
2904 \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.

```
2905 \ifglsxindy
```

```

2906 \newcommand*{\GlsSetXdyMinRangeLength}[1]{%
2907 \renewcommand*{\@glsminrange}{#1}}
2908 \else
2909 \newcommand*{\GlsSetXdyMinRangeLength}[1]{%
2910 \glsnoxindywarning\GlsSetXdyMinRangeLength}
2911 \fi

```

\writeist

```

2912 \ifglxindy
    Code to use if xindy is required.
2913 \def\writeist{%
    Update attributes list
2914 \@gls@addpredefinedattributes
    Open the file.
2915 \openout\glswrite=\istfilename
    Write header comment at the start of the file
2916 \write\glswrite{;; xindy style file created by the glossaries
2917 package}%
2918 \write\glswrite{;; for document '\jobname' on
2919 \the\year-\the\month-\the\day}%
    Specify the required styles
2920 \write\glswrite{^^J; required styles^^J}
2921 \@for\@xdystyle:=\@xdyrequiredstyles\do{%
2922 \ifx\@xdystyle\@empty
2923 \else
2924 \protected@write\glswrite{{(require
2925 \string"\@xdystyle.xdy\string")}}%
2926 \fi
2927 }%
    List the allowed attributes (possible values used by the format key)
2928 \write\glswrite{^^J%
2929 ; list of allowed attributes (number formats)^^J}%
2930 \write\glswrite{(define-attributes ((\@xdyattributes)))}%
    Define any additional alphabets
2931 \write\glswrite{^^J; user defined alphabets^^J}%
2932 \write\glswrite{\@xdyuseralphabets}%
    Define location classes.
2933 \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.
2934 \@for\@gls@classI:=\@gls@xdy@locationlist\do{%

```

Case were $\langle Hprefix \rangle$ is empty:

```

2935 \protected@write\glswrite{}\{(define-location-class
2936 \string"\@gls@classI\string"^^J\space\space\space
2937 (
2938 :sep "{ }{"
2939 \csname @gls@xdy@Lclass@\@gls@classI\endcsname\space
2940 :sep "}"
2941 )
2942 ^^J\space\space\space
2943 :min-range-length \@glsminrange^^J%
2944 )
2945 }%
```

Nested iteration over all classes:

```

2946 {%
2947 \@for\@gls@classII:=\@gls@xdy@locationlist\do{%
2948 \protected@write\glswrite{}\{(define-location-class
2949 \string"\@gls@classII-\@gls@classI\string"
2950 ^^J\space\space\space
2951 (
2952 :sep "{"
2953 \csname @gls@xdy@Lclass@\@gls@classII\endcsname\space
2954 :sep "}"
2955 \csname @gls@xdy@Lclass@\@gls@classI\endcsname\space
2956 :sep "}"
2957 )
2958 ^^J\space\space\space
2959 :min-range-length \@glsminrange^^J%
2960 )
2961 }%
2962 }%
2963 }%
2964 }%
```

User defined location classes (needs checking for new location format).

```

2965 \write\glswrite{^^J; user defined location classes}%
2966 \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.)

```

2967 \write\glswrite{^^J; define cross-reference class^^J}%
2968 \write\glswrite{(define-crossref-class \string"see\string"
2969 :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.)

```

2970 \write\glswrite{(markup-crossref-list
```

```

2971      :class \string"see\string"^^J\space\space\space
2972      :open \string"\string\glsseeformat\string"
2973      :close \string"{}\string")}%

```

List the order to sort the classes.

```

2974      \write\glswrite{^^J; define the order of the location classes}%
2975      \write\glswrite{(define-location-class-order
2976      (\@xdylocationclassorder))}%

```

Specify what to write to the start and end of the glossary file.

```

2977      \write\glswrite{^^J; define the glossary markup^^J}%

2978      \write\glswrite{(markup-index^^J\space\space\space
2979      :open \string"\string
2980      \glossarysection[\string\glossarytoctitle]{\string
2981      \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)

```

2982      \@for\@this@ctr:=\@xdycounters\do{%
2983      {%
2984      \@for\@this@attr:=\@xdyattributelist\do{%
2985      \protected\write\glswrite{{}\string\providecommand*%
2986      \expandafter\string
2987      \csname glsX\@this@ctr X\@this@attr\endcsname[2]%
2988      {%
2989      \string\setentrycounter
2990      [\expandafter\@gobble\string\#1]{\@this@ctr}%
2991      \expandafter\string
2992      \csname\@this@attr\endcsname
2993      {\expandafter\@gobble\string\#2}%
2994      }%
2995      }%
2996      }%
2997      }%
2998      }%

```

Add the end part of the open tag and the rest of the markup-index information:

```

2999      \write\glswrite{%
3000      \string\begin
3001      {theglossary}\string\glossaryheader\string~n\string" ^^J\space
3002      \space\space:close \string"\expandafter\@gobble
3003      \string%\string~n\string
3004      \end{theglossary}\string\glossarypostamble
3005      \string~n\string" ^^J\space\space\space
3006      :tree)}}%

```

Specify what to put between letter groups

```

3007      \write\glswrite{(markup-letter-group-list
3008      :sep \string"\string\glsgroupskip\string~n\string")}%

```

Specify what to put between entries

```
3009 \write\glswrite{(markup-indexentry
3010 :open \string"\string\relax \string\glsresetentrylist
3011 \string~n\string")}%
```

Specify how to format entries

```
3012 \write\glswrite{(markup-locclass-list :open
3013 \string"\glsopenbrace\string\glossaryentrynumbers
3014 \glsopenbrace\string\relax\space \string"^^J\space\space\space
3015 :sep \string", \string"
3016 :close \string"\glsclosebrace\glsclosebrace\string")}%
```

Specify how to separate location numbers

```
3017 \write\glswrite{(markup-locref-list
3018 :sep \string"\string\delimN\space\string")}%
```

Specify how to indicate location ranges

```
3019 \write\glswrite{(markup-range
3020 :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.

```
3021 \@onelevel@sanitize\gls@suffixF
3022 \@onelevel@sanitize\gls@suffixFF
3023 \ifx\gls@suffixF\@empty
3024 \else
3025 \write\glswrite{(markup-range
3026 :close "\gls@suffixF" :length 1 :ignore-end)}%
3027 \fi
3028 \ifx\gls@suffixFF\@empty
3029 \else
3030 \write\glswrite{(markup-range
3031 :close "\gls@suffixFF" :length 2 :ignore-end)}%
3032 \fi
```

Specify how to format locations.

```
3033 \write\glswrite{^^J; define format to use for locations^^J}%
3034 \write\glswrite{\@xdylocref}%
```

Specify how to separate letter groups.

```
3035 \write\glswrite{^^J; define letter group list format^^J}%
3036 \write\glswrite{(markup-letter-group-list
3037 :sep \string"\string\glsgroupskip\string~n\string")}%
```

Define letter group headings.

```
3038 \write\glswrite{^^J; letter group headings^^J}%
3039 \write\glswrite{(markup-letter-group
3040 :open-head \string"\string\glsgroupheading
3041 \glsopenbrace\string"^^J\space\space\space
3042 :close-head \string"\glsclosebrace\string")}%
```

Define additional letter groups.

```
3043 \write\glswrite{^^J; additional letter groups^^J}%
3044 \write\glswrite{\@xdylettergroups}%
```

Define additional sort rules

```
3045 \write\glswrite{^^J; additional sort rules^^J}
3046 \write\glswrite{\@xdysortrules}%
```

Close the style file

```
3047 \closeout\glswrite
```

Suppress any further calls.

```
3048 \let\writeist\relax
3049 }
3050 \else
```

Code to use if makeindex is required.

```
3051 \edef\@gls@actualchar{\string?}
3052 \edef\@gls@encapchar{\string|}
3053 \edef\@gls@levelchar{\string!}
3054 \edef\@gls@quotechar{\string"}
3055 \def\writeist{\relax
3056 \openout\glswrite=\istfilename
3057 \write\glswrite{\expandafter\@gobble\string\% makeindex style file
3058 created by the glossaries package}
3059 \write\glswrite{\expandafter\@gobble\string\% for document
3060 '\jobname' on \the\year-\the\month-\the\day}
3061 \write\glswrite{actual '\@gls@actualchar'}
3062 \write\glswrite{encap '\@gls@encapchar'}
3063 \write\glswrite{level '\@gls@levelchar'}
3064 \write\glswrite{quote '\@gls@quotechar'}
3065 \write\glswrite{keyword \string"\string\glossaryentry\string"}
3066 \write\glswrite{preamble \string"\string\glossarysection[\string
3067 \glossarytoctitle]{\string\glossarytitle}\string
3068 \glossarypreamble\string\n\string\begin{theglossary}\string
3069 \glossaryheader\string\n\string"}
3070 \write\glswrite{postamble \string"\string%\string\n\string
3071 \end{theglossary}\string\glossarypostamble\string\n
3072 \string"}
3073 \write\glswrite{group_skip \string"\string\glsgroupskip\string\n
3074 \string"}
3075 \write\glswrite{item_0 \string"\string%\string\n\string"}
3076 \write\glswrite{item_1 \string"\string%\string\n\string"}
3077 \write\glswrite{item_2 \string"\string%\string\n\string"}
3078 \write\glswrite{item_01 \string"\string%\string\n\string"}
3079 \write\glswrite{item_x1
3080 \string"\string\relax \string\glsresetentrylist\string\n
3081 \string"}
3082 \write\glswrite{item_12 \string"\string%\string\n\string"}
3083 \write\glswrite{item_x2
3084 \string"\string\relax \string\glsresetentrylist\string\n
```

```

3085     \string"}

3086     \write\glswrite{delim_0 \string"\string\{\string
3087         \glossaryentrynumbers\string\{\string\relax \string"}
3088     \write\glswrite{delim_1 \string"\string\{\string
3089         \glossaryentrynumbers\string\{\string\relax \string"}
3090     \write\glswrite{delim_2 \string"\string\{\string
3091         \glossaryentrynumbers\string\{\string\relax \string"}
3092     \write\glswrite{delim_t \string"\string\}\string\}\string"}
3093     \write\glswrite{delim_n \string"\string\delimN \string"}
3094     \write\glswrite{delim_r \string"\string\delimR \string"}
3095     \write\glswrite{headings_flag 1}
3096     \write\glswrite{heading_prefix
3097         \string"\string\glsgroupheading\string\{\string"}
3098     \write\glswrite{heading_suffix
3099         \string"\string\}\string\relax
3100         \string\glsgroupresetentrylist \string"}
3101     \write\glswrite{symhead_positive \string"glssymbols\string"}
3102     \write\glswrite{numhead_positive \string"glslnumbers\string"}
3103     \write\glswrite{page_compositor \string"glscpositor\string"}
3104     \@gls@escbsdq\gls@suffixF
3105     \@gls@escbsdq\gls@suffixFF
3106     \ifx\gls@suffixF\@empty
3107     \else
3108         \write\glswrite{suffix_2p \string"\gls@suffixF\string"}
3109     \fi
3110     \ifx\gls@suffixFF\@empty
3111     \else
3112         \write\glswrite{suffix_3p \string"\gls@suffixFF\string"}
3113     \fi
3114     \closeout\glswrite
3115     \let\writeist\relax
3116 }
3117 \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`

```

3118 \newcommand{\noist}{%
    Update attributes list
3119     \@gls@addpredefinedattributes
3120     \let\writeist\relax
3121 }

```

`\@makeglossary` is an internal command that takes an argument indicating the glossary type. This command will create the glossary file required by `makeindex` for the given glossary type, using the extension supplied by

the `<out-ext>` parameter used in `\newglossary` (and it will also activate the `\glossary` command, and create the customized .ist makeindex style file).

Note that you can't use `\makeglossary` for only some of the defined glossaries. You either need to have a `\makeglossary` for all glossaries or none (otherwise you will end up with a situation where \TeX is trying to write to a non-existent file). The relevant glossary must be defined prior to using `\makeglossary`.

`\@makeglossary`

```
3122 \newcommand*{\@makeglossary}[1]{%
3123   \ifglossaryexists{#1}%
3124   {%
```

Only create a new write if `savewrites=false` otherwise create a token to collect the information.

```
3125   \ifglssavewrites
3126     \expandafter\newtoks\csname glo@#1@filetok\endcsname
3127   \else
3128     \expandafter\newwrite\csname glo@#1@file\endcsname
3129     \expandafter\@glsopenfile\csname glo@#1@file\endcsname{#1}%
3130   \fi
3131   \@gls@renewglossary
3132   \writeist
3133 }%
3134 {%
3135   \PackageError{glossaries}%
3136   {Glossary type ‘#1’ not defined}%
3137   {New glossaries must be defined before using \string\makeglossary}%
3138 }%
3139 }
```

`\@glsopenfile` Open write file associated with the given glossary.

```
3140 \newcommand*{\@glsopenfile}[2]{%
3141   \immediate\openout#1=\jobname.\csname @glotype@#2@out\endcsname
3142   \PackageInfo{glossaries}{Writing glossary file
3143     \jobname.\csname @glotype@#2@out\endcsname}%
3144 }
```

`\@nomakeglossaries` Issue warning that `\makeglossaries` hasn't been used.

```
3145 \newcommand*{\@warn@nomakeglossaries}{%
3146   \GlossariesWarningNoLine{\string\makeglossaries\space
3147   hasn't been used,^^Jthe glossaries will not be updated}%
3148 }
```

`\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`

```
3149 \newcommand*\makeglossaries{%
```

Write the name of the style file to the aux file (needed by `makeglossaries`)

```
3150 \protected@write\@auxout{}\string\@istfilename{\istfilename}}%
```

```
3151 \protected@write\@auxout{}\string\@glorder{\glorder}}
```

Iterate through each glossary type and activate it.

```
3152 \@for\@glo@type:=\@glo@types\do{%
```

```
3153 \ifthenelse{\equal{\@glo@type}{}}{\}%
```

```
3154 \@makeglossary{\@glo@type}}%
```

```
3155 }%
```

New glossaries must be created before `\makeglossaries` so disable `\newglossary`.

```
3156 \renewcommand*\newglossary[4][]{%
```

```
3157 \PackageError{glossaries}{New glossaries
```

```
3158 must be created before \string\makeglossaries}{You need
```

```
3159 to move \string\makeglossaries\space after all your
```

```
3160 \string\newglossary\space commands}}%
```

Any subsequent instances of this command should have no effect

```
3161 \let\@makeglossary\relax
```

```
3162 \let\makeglossary\relax
```

```
3163 \let\makeglossaries\relax
```

Disable all commands that have no effect after `\makeglossaries`

```
3164 \@disable@onlypremakeg
```

Suppress warning about no `\makeglossaries`

```
3165 \let\warn@nomakeglossaries\relax
```

Declare list parser for `\glsdisplaynumberlist`

```
3166 \ifglssavenumberlist
```

```
3167 \edef\@gls@dodolistparser{\noexpand\DeclareListParser
```

```
3168 {\noexpand\glsnumlistparser}{\delimN}}%
```

```
3169 \@gls@dodolistparser
```

```
3170 \fi
```

```
3171 }
```

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`

```
3172 \let\makeglossary\makeglossaries
```

If `\makeglossaries` hasn't been used, issue a warning. Also issue a warning if neither `\printglossaries` nor `\printglossary` have been used.

```
3173 \AtEndDocument{%
```

```
3174 \warn@nomakeglossaries
```

```
3175 \warn@noprintglossary
```

```
3176 }
```

1.13 Writing information to associated files

`\glswrite` The write used for style file also used for all other output files if `savewrites=true`.

```
3177 \newwrite\glswrite
```

`\istfile` Deprecated.

```
3178 \def\istfile{\glswrite}
```

At the end of the document, the files should be created if `savewrites=true`.

```
3179 \AtEndDocument{%
```

```
3180   \glswritefiles
```

```
3181 }
```

`\glswritefiles` Only write the files if `savewrites=true`

```
3182 \ifglssavewrites
```

```
3183   \newcommand*{\glswritefiles}{%
```

Iterate through all the glossaries

```
3184   \foralllglossaries{\@glo@type}{%
```

Check for empty glossaries (patch provided by Patrick Häcker)

```
3185     \ifcsundef{glo@\@glo@type @filetok}%
```

```
3186     {%
```

```
3187       \def\gls@tmp{}%
```

```
3188     }%
```

```
3189     {%
```

```
3190       \edef\gls@tmp{\expandafter\the
```

```
3191         \csname glo@\@glo@type @filetok\endcsname}%
```

```
3192     }%
```

```
3193     \ifx\gls@tmp\@empty
```

```
3194       \ifx\@glo@type\glsdefaulttype
```

```
3195         \GlossariesWarningNoLine{Glossary ‘\@glo@type’ has no
```

```
3196           entries.^^JRemember to use package option ‘nomain’ if
```

```
3197 you
```

```
3198           don’t want to^^Juse the main glossary}%
```

```
3199       \else
```

```
3200         \GlossariesWarningNoLine{Glossary ‘\@glo@type’ has no
```

```
3201           entries}%
```

```
3202       \fi
```

```
3203     \else
```

```
3204       \@glsopenfile{\glswrite}{\@glo@type}%
```

```
3205       \immediate\write\glswrite{%
```

```
3206         \expandafter\the
```

```
3207         \csname glo@\@glo@type @filetok\endcsname}%
```

```
3208       \immediate\closeout\glswrite
```

```
3209     \fi
```

```
3210   }%
```

```
3211 }
```

```
3212 \else
```

```
3213   \let\glswritefiles\relax
```

3214 \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`

```
3215 \renewcommand*{\glossary}[1][\glsdefaulttype]{%
3216   \@glossary[#1]%
3217 }
```

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`

```
3218 \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`

```
3219 \newcommand{\@gls@renewglossary}{%
3220   \gdef\@glossary[##1]{\@bsphack\begingroup\@wrglossary{##1}}%
3221   \let\@gls@renewglossary\@empty
3222 }
```

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`

```
3223 \renewcommand*{\@wrglossary}[2]{%
3224   \ifglssavewrites
3225     \protected@edef\@gls@tmp{\the\csname glo@#1@filetok\endcsname#2}%
3226     \expandafter\global\expandafter\csname glo@#1@filetok\endcsname
3227       \expandafter{\@gls@tmp^^J}%
3228   \else
3229     \ifcsdef{glo@#1@file}%
3230       {%
3231         \expandafter\protected@write\csname glo@#1@file\endcsname{}{#2}%
3232       }%
3233     {%
3234       \GlossariesWarning{No file defined for glossary ‘#1’}%
3235     }%
3236   \fi
3237   \endgroup\@esphack
3238 }
```

\do@wrglossary

```
3239 \newcommand*{\do@wrglossary}[1]{%
3240   \ifglsonlyfirst
3241     \ifglused{#1}{\do@wrglossary{#1}}%
3242   \else
3243     \do@wrglossary{#1}%
3244   \fi
3245 }
```

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.

```
3246 \newcommand*{\do@wrglossary}[1]{%
```

Get the location and escape any special characters

```
3247   \protected@edef\glslocref{\theglentrycounter}%
3248   \gls@checkmkidxchars\glslocref
```

Check if the hyper-location is the same as the location and set the hyper prefix.

```
3249   \expandafter\ifx\theglentrycounter\theglentrycounter
3250     \def\glo@counterprefix{%
3251   \else
3252     \protected@edef\glsHlocref{\theglentrycounter}%
3253     \gls@checkmkidxchars\glsHlocref
3254     \edef\do@gls@getcounterprefix{\noexpand\gls@getcounterprefix
3255       {\glslocref}{\glsHlocref}}%
3256     }%
3257     \do@gls@getcounterprefix
3258   \fi
```

Determine whether to use xindy or makeindex syntax

```
3259   \ifglxindy
```

Need to determine if the formatting information starts with a (or) indicating a range.

```
3260     \expandafter\glo@check@mkidxrangechar\glsnumberformat\@nil
3261     \def\glo@range{%
3262   \expandafter\if\glo@prefix(\relax
3263     \def\glo@range{:open-range}%
3264   \else
3265     \expandafter\if\glo@prefix)\relax
3266     \def\glo@range{:close-range}%
3267   \fi
3268   \fi
```

Write to the glossary file using xindy syntax.

```
3269     \glossary[\csname glo@#1@type\endcsname]{%
3270       (indexentry :key (\csname glo@#1@index\endcsname)
3271         :locref \string{\glo@counterprefix}{\theglentrycounter}\string" %
3272         :attr \string\gls@counter\glo@suffix\string"
```

```

3273     \@glo@range
3274   )
3275   }%
3276   \else

```

Convert the format information into the format required for makeindex

```

3277   \@set@glo@numformat{\@glo@numfmt}{\@gls@counter}{\@glsnumberformat}%
3278   {\@glo@counterprefix}%

```

Write to the glossary file using makeindex syntax.

```

3279   \glossary[\csname glo@#1@type\endcsname]{%
3280   \string\glossaryentry{\csname glo@#1@index\endcsname
3281   \@gls@encapchar\@glo@numfmt}{\theglsentrycounter}}%
3282   \fi
3283 }

```

`\ls@getcounterprefix` Get the prefix that needs to be prepended to counter in order to get the hyper counter. (For example, with the standard article class and hyperref, `\theequation` needs to be prefixed with `\section num` to get the equivalent `\theHequation`.) NB this assumes that the prefix ends with a dot, which is the standard. (Otherwise it makes the xindy location classes more complicated.)

```

3284 \newcommand*\@gls@getcounterprefix[2]{%
3285   \edef\@gls@thisloc{#1}\edef\@gls@thisHloc{#2}%
3286   \ifx\@gls@thisloc\@gls@thisHloc
3287     \def\@glo@counterprefix{%
3288     \else
3289     \def\@gls@get@counterprefix##1.#1##2\end@getprefix{%
3290       \def\@glo@tmp{##2}%
3291       \ifx\@glo@tmp\@empty
3292         \def\@glo@counterprefix{%
3293       \else
3294         \def\@glo@counterprefix{##1}%
3295       \fi
3296     }%
3297     \@gls@get@counterprefix#2.#1\end@getprefix
3298   \fi
3299 }

```

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.

```

3300 \newcommand{\@do@seeglossary}[2]{%
3301 \def\@gls@xref{#2}%
3302 \@onelevel@sanitize\@gls@xref
3303 \@gls@checkmkidxchars\@gls@xref
3304 \ifglsxindy
3305   \glossary[\csname glo@#1@type\endcsname]{%

```

```

3306      (indexentry
3307        :tkey (\csname glo@#1@index\endcsname)
3308        :xref (\string"\@gls@xref\string")
3309        :attr \string"see\string"
3310      )
3311    }%
3312  \else
3313    \glossary[\csname glo@#1@type\endcsname]{%
3314      \string\glossaryentry{\csname glo@#1@index\endcsname
3315        \@gls@encapchar glsseeformat\@gls@xref}{Z}}%
3316  \fi
3317 }

```

`\@gls@fixbraces` If no optional argument is specified, list needs to be enclosed in a set of braces.

```

3318 \def\@gls@fixbraces#1#2#3\@nil{%
3319   \ifx#2[\relax
3320     \def#1{#2#3}%
3321   \else
3322     \def#1{{#2#3}}%
3323   \fi
3324 }

```

`\glssee` `\glssee{<label>}{<cross-ref list>}`

```

3325 \newcommand*{\glssee}[3][\seename]{%
3326   \@do@seeglossary{#2}{[ #1]{#3}}
3327 \newcommand*{\@glssee}[3][\seename]{%
3328   \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.

```

3329 \newcommand*{\glsseeformat}[3][\seename]{\emph{#1} \glsseelist{#2}}

```

`\glsseelist` `\glsseelist{<list>}` formats list of entry labels.

```

3330 \newcommand*{\glsseelist}[1]{%

```

If there is only one item in the list, set the last separator to do nothing.

```

3331 \let\@gls@dolast\relax

```

Don’t display separator on the first iteration of the loop

```

3332 \let\@gls@donext\relax

```

Iterate through the labels

```

3333 \@for\@gls@thislabel:=#1\do{%

```

Check if on last iteration of loop

```

3334   \ifx\@xfor@nextelement\@nnil
3335     \@gls@dolast
3336   \else
3337     \@gls@donext
3338   \fi

```

```

display the entry for this label
3339 \glsseeitem{\@gls@thislabel}%
Update separators
3340 \let\@gls@dolast\glsseelastsep
3341 \let\@gls@donext\glsseesep
3342 }%
3343 }

```

`\glsseelastsep` Separator to use between penultimate and ultimate entries in a cross-referencing list.

```
3344 \newcommand*{\glsseelastsep}{\space\andname\space}
```

`\glsseesep` Separator to use between entires in a cross-referencing list.

```
3345 \newcommand*{\glsseesep}{, }
```

`\glsseeitem` `\glsseeitem{<label>}` formats individual entry in a cross-referencing list.

```
3346 \newcommand*{\glsseeitem}[1]{\gls hyperlink[\glsseeitemformat{#1}]{#1}}
```

`\glsseeitemformat` As from v3.0, default is to use `\glsentrytext` instead of `\glsentryname`. (To avoid problems with the name key being sanitized.)

```
3347 \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.

```

3348 \newcommand*{\gls@save@numberlist}[1]{%
3349 \ifglssavenumberlist
3350 \toks@{#1}%
3351 \edef\@do@writeaux@info{%
3352 \noexpand\csgdef{glo@\glscurrententrylabel @numberlist}{\the\toks@}%
3353 }%
3354 \@onelevel@sanitize\@do@writeaux@info
3355 \protected@write\@auxout{}\@do@writeaux@info}%
3356 \fi
3357 }

```

`\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.)

```

3358 \def\warn@noprintglossary{%
3359   \GlossariesWarningNoLine{No \string\printglossary\space
3360     or \string\printglossaries\space
3361     found.^^JThis document will not have a glossary}%
3362 }

```

`\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.

```

3363 \ifcsundef{printglossary}{}%
3364 {%

```

If `\printglossary` is already defined, issue a warning and undefine it.

```

3365   \GlossariesWarning{Overriding \string\printglossary}%
3366   \undef\printglossary
3367 }

```

`\printglossary` has an optional argument. The default value is to set the glossary type to the main glossary.

```

3368 \newcommand*{\printglossary}[1][type=\glsdefaulttype]{%

```

If xindy is being used, need to find the root language for makeglossaries to pass to xindy.

```

3369   \ifglsxindy\findrootlanguage\fi

```

Set up defaults.

```

3370   \def\@glo@type{\glsdefaulttype}%
3371   \def\glossarytitle{\csname @glo@type\@glo@type @title\endcsname}%

3372   \def\glossarytoctitle{\glossarytitle}%
3373   \let\org@glossarytitle\glossarytitle
3374   \def\@glossarystyle{}%
3375   \def\gls@dotoc@title{\glssettoctitle{\@glo@type}}%

```

Store current value of `\glossaryentrynumbers`. (This may be changed via the optional argument)

```

3376   \let\org@glossaryentrynumbers\glossaryentrynumbers

```

Localise the effects of the optional argument

```

3377   \bgroup

```

Determine settings specified in the optional argument.

```

3378   \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)

```

3379   \ifx\glossarytitle\org@glossarytitle
3380   \else
3381     \expandafter\let\csname @glo@type\@glo@type @title\endcsname
3382       \glossarytitle
3383   \fi

```

Allow a high-level user command to indicate the current glossary

```

3384   \let\currentglossary\@glo@type

```


Enable individual number lists to be suppressed.

```
3385 \let\org@glossaryentrynumbers\glossaryentrynumbers
3386 \let\glsnonextpages\@glsnonextpages
```

Enable individual number list to be activated:

```
3387 \let\glsnextpages\@glsnextpages
```

Enable suppression of description terminators.

```
3388 \let\nopostdesc\@nopostdesc
```

Set up the entry for the TOC

```
3389 \gls@dotoc@title
```

Set the glossary style

```
3390 \@glossarystyle
```

added a way to fetch the current entry label:

```
3391 \let\gls@org@glossaryentryfield\glossaryentryfield
3392 \let\gls@org@glossarysubentryfield\glossarysubentryfield
3393 \renewcommand{\glossaryentryfield}[1]{%
3394   \gdef\glscurrententrylabel{##1}%
3395   \gls@org@glossaryentryfield{##1}%
3396 }%
3397 \renewcommand{\glossarysubentryfield}[2]{%
3398   \gdef\glscurrententrylabel{##2}%
3399   \gls@org@glossarysubentryfield{##1}{##2}%
3400 }%
```

Some macros may end up being expanded into internals in the glossary, so need to make @ a letter.

```
3401 \makeatletter
```

Input the glossary file, if it exists.

```
3402 \@input@{\jobname.\csname @glo@type@\@glo@type @in\endcsname}%
```

If the glossary file doesn't exist, do \null. (This ensures that the page is shipped out and all write commands are done.) This might produce an empty page, but at this point the document isn't complete, so it shouldn't matter.

```
3403 \IfFileExists{\jobname.\csname @glo@type@\@glo@type @in\endcsname}%
3404 {}%
3405 {\null}%
```

If xindy is being used, need to write the language dependent information to the .aux file for makeglossaries.

```
3406 \ifglxindy
3407   \ifcsundef{@xdy@\@glo@type @language}%
3408   {%
3409     \protected@write\@auxout{}{%
3410       \string\@xdy@language{\@glo@type}{\@xdy@main@language}}%
3411   }%
3412   {%
3413     \protected@write\@auxout{}{%
```

```

3414         \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type
3415             @language\endcsname}}}%
3416     }%
3417     \protected@write\@auxout{}{%
3418         \string\@gls@codepage{\@glo@type}{\@gls@codepage}}}%
3419     \fi
3420 \egroup

Reset \glossaryentrynumbers
3421 \global\let\glossaryentrynumbers\@org@glossaryentrynumbers

Suppress warning about no \printglossary
3422 \global\let\warn@noprintglossary\relax
3423 }

```

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`

```

3424 \newcommand*{\printglossaries}{%
3425   \forallglossaries{\@glo@type}{\printglossary[type=\@glo@type]}}%
3426 }

```

The keys that can be used in the optional argument to `\printglossary` are as follows: The `type` key sets the glossary type.

```

3427 \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`.

```

3428 \define@key{printgloss}{title}{%
3429   \def\glossarytitle{#1}%
3430   \let\gls@dotoc@title\relax
3431 }

```

The `toctitle` sets the text used for the relevant entry in the table of contents.

```

3432 \define@key{printgloss}{toctitle}{%
3433   \def\glossarytoctitle{#1}%
3434   \let\gls@dotoc@title\relax
3435 }

```

The `style` key sets the glossary style (but only for the given glossary).

```

3436 \define@key{printgloss}{style}{%
3437   \ifcsundef{\glsstyle@#1}%
3438   {%

```

```

3439 \PackageError{glossaries}%
3440 {Glossary style ‘#1’ undefined}{}%
3441 }%
3442 {%
3443 \def\@glossarystyle{\csname @glsstyle@#1\endcsname}%
3444 }%
3445 }

```

The `numberedsection` key determines if this glossary should be in a numbered section.

```

3446 \define@choicekey{printgloss}{numberedsection}[\val\nr]{%
3447 false,nolabel,autolabel}[nolabel]{%
3448 \ifcase\nr\relax
3449 \renewcommand*\@glossarysecstar{*}%
3450 \renewcommand*\@glossaryseclabel{}%
3451 \or
3452 \renewcommand*\@glossarysecstar{}%
3453 \renewcommand*\@glossaryseclabel{}%
3454 \or
3455 \renewcommand*\@glossarysecstar{}%
3456 \renewcommand*\@glossaryseclabel{\label{\glsautoprefix\@glo@type}}%
3457 \fi}

```

The `nonumberlist` key determines if this glossary should have a number list.

```

3458 \define@boolkey{printgloss}[gls]{nonumberlist}[true]{%
3459 \ifglsonumberlist
3460 \def\glossaryentrynumbers##1{}%
3461 \else
3462 \def\glossaryentrynumbers##1{##1}%
3463 \fi}

```

`\@glsonextpages` 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 `\glsonextpages` 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.

```

3464 \newcommand*\@glsonextpages{%
3465 \gdef\glossaryentrynumbers##1{%
3466 \glsresetentrylist
3467 }%
3468 }

```

`\@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.

```

3469 \newcommand*\@glsnextpages{%
3470 \gdef\glossaryentrynumbers##1{%

```

```

3471      ##1\glsresetentrylist}}

\glsresetentrylist  Resets \glossaryentrynumbers
3472 \newcommand*{\glsresetentrylist}{%
3473   \global\let\glossaryentrynumbers\org@glossaryentrynumbers}

\glsnonextpages  Outside of \printglossary this does nothing.
3474 \newcommand*{\glsnonextpages}{}

\glsnextpages  Outside of \printglossary this does nothing.
3475 \newcommand*{\glsnextpages}{}

glossaryentry  If the entrycounter package option has been used, define a counter to number
               each level 0 entry.
3476 \ifglentrycounter
3477   \ifx\@gls@counterwithin\@empty
3478     \newcounter{glossaryentry}
3479   \else
3480     \newcounter{glossaryentry}[\@gls@counterwithin]
3481   \fi
3482   \def\theHglossaryentry{\currentglossary.\theglossaryentry}
3483 \fi

glossarysubentry  If the subentrycounter package option has been used, define a counter to num-
                  ber each level 1 entry.
3484 \ifglssubentrycounter
3485   \ifglentrycounter
3486     \newcounter{glossarysubentry}[glossaryentry]
3487   \else
3488     \newcounter{glossarysubentry}
3489   \fi
3490   \def\theHglossarysubentry{\currentglssubentry.\theglossarysubentry}
3491 \fi

esetsubentrycounter  Resets the glossarysubentry counter.
3492 \ifglssubentrycounter
3493   \newcommand*{\glsresetsubentrycounter}{%
3494     \setcounter{glossarysubentry}{0}%
3495   }
3496 \else
3497   \newcommand*{\glsresetsubentrycounter}{}
3498 \fi

esetsubentrycounter  Resets the glossareentry counter.
3499 \ifglentrycounter
3500   \newcommand*{\glsresetentrycounter}{%
3501     \setcounter{glossaryentry}{0}%
3502   }

```

```

3503 \else
3504   \newcommand*{\glsresetentrycounter}{}
3505 \fi

```

`\glsstepentry` Advance the glossaryentry counter if in use. The argument is the label associated with the entry.

```

3506 \ifglentrycounter
3507   \newcommand*{\glsstepentry}[1]{%
3508     \refstepcounter{glossaryentry}%
3509     \label{glsentry-#1}%
3510   }
3511 \else
3512   \newcommand*{\glsstepentry}[1]{}
3513 \fi

```

`\glsstepsubentry` Advance the glossarysubentry counter if in use. The argument is the label associated with the subentry.

```

3514 \ifglssubentrycounter
3515   \newcommand*{\glsstepsubentry}[1]{%
3516     \def\currentglssubentry{#1}%
3517     \refstepcounter{glossarysubentry}%
3518     \label{glsentry-#1}%
3519   }
3520 \else
3521   \newcommand*{\glsstepsubentry}[1]{}
3522 \fi

```

`\glsrefentry` Reference the entry or sub-entry counter if in use, otherwise just do `\gls`.

```

3523 \ifglentrycounter
3524   \newcommand*{\glsrefentry}[1]{\ref{glsentry-#1}}
3525 \else
3526   \ifglssubentrycounter
3527     \newcommand*{\glsrefentry}[1]{\ref{glsentry-#1}}
3528   \else
3529     \newcommand*{\glsrefentry}[1]{\gls{#1}}
3530   \fi
3531 \fi

```

`glentrycounterlabel` Defines how to display the glossaryentry counter.

```

3532 \ifglentrycounter
3533   \newcommand*{\glentrycounterlabel}{\theglossaryentry.\space}
3534 \else
3535   \newcommand*{\glentrycounterlabel}{}
3536 \fi

```

`glsubentrycounterlabel` Defines how to display the glossarysubentry counter.

```

3537 \ifglssubentrycounter
3538   \newcommand*{\glsubentrycounterlabel}{\theglossarysubentry)\space}

```

```

3539 \else
3540   \newcommand*{\glssubentrycounterlabel}{}
3541 \fi

```

`\glentryitem` Step and display glossaryentry counter, if appropriate.

```

3542 \ifglentrycounter
3543   \newcommand*{\glentryitem}[1]{%
3544     \glstepentry{#1}\glentrycounterlabel
3545   }
3546 \else
3547   \newcommand*{\glentryitem}[1]{\glresetsubentrycounter}
3548 \fi

```

`\glssubentryitem` Step and display glossarysubentry counter, if appropriate.

```

3549 \ifglssubentrycounter
3550   \newcommand*{\glssubentryitem}[1]{%
3551     \glstepsubentry{#1}\glssubentrycounterlabel
3552   }
3553 \else
3554   \newcommand*{\glssubentryitem}[1]{}
3555 \fi

```

`theglossary` If the `theglossary` environment has already been defined, a warning will be issued. This environment should be redefined by glossary styles.

```

3556 \ifcsundef{theglossary}%
3557 {%
3558   \newenvironment{theglossary}{}{}%
3559 }%
3560 {%
3561   \GlossariesWarning{overriding ‘theglossary’ environment}%
3562   \renewenvironment{theglossary}{}{}%
3563 }

```

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`

```

3564 \newcommand*{\glossaryheader}{}

```

`\glstarget` `\glstarget{<label>}{<name>}`

Provide user interface to `\@glstarget` to make it easier to modify the glossary style in the document.

```

3565 \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>`.

```
3566 \newcommand*{\glossaryentryfield}[5]{%
3567 \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.)

```
3568 \newcommand*{\glossarysubentryfield}[6]{%
3569 \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`

```
3570 \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`, `glsglnumbers`, A, ..., Z. Glossary styles must redefine this command. (In between groups, `\glsgroupheading` comes immediately after `\glsgroupskip`.)

`\glsgroupheading`

```
3571 \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 `\glsgrouptitle` and `\glsgrouplabel` 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`

```
3572 \newcommand*{\glsgetgrouptitle}[1]{%
3573   \ifcsundef{#1groupname}{#1}{\csname #1groupname\endcsname}%
3574 }
```

`\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`

```
3575 \newcommand*{\glsgetgrouplabel}[1]{%
3576   \ifthenelse{\equals{#1}{\glssymbolsgroupname}}{\glssymbols}{%
3577   \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`

```
3578 \newcommand*{\setentrycounter}[2][ ]{%
3579   \def\@glo@counterprefix{#1}%
3580   \ifx\@glo@counterprefix\@empty
3581     \def\@glo@counterprefix{.}%
3582   \else
3583     \def\@glo@counterprefix{.#1}%
3584   \fi
3585   \def\glsentrycounter{#2}%
3586 }
```

The current glossary style can be set using `\glossarystyle{<style>}`.

`\glossarystyle`

```
3587 \newcommand*{\glossarystyle}[1]{%
3588   \ifcsundef{@glsstyle@#1}%
3589   {%
3590     \PackageError{glossaries}{Glossary style ‘#1’ undefined}{%}
```



```

3591 }%
3592 {%
3593     \csname @glsstyle@#1\endcsname
3594 }%
3595 }

```

`\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.

```

3596 \newcommand{\newglossarystyle}[2]{%
3597     \ifcsundef{@glsstyle@#1}%
3598     {%
3599         \expandafter\def\csname @glsstyle@#1\endcsname{#2}%
3600     }%
3601     {%
3602         \PackageError{glossaries}{Glossary style ‘#1’ is already defined}{}%
3603     }%
3604 }

```

`\renewglossarystyle` Code for this macro supplied by Marco Daniel.

```

3605 \newcommand{\renewglossarystyle}[2]{%
3606     \ifcsundef{@glsstyle@#1}%
3607     {%
3608         \PackageError{glossaries}{Glossary style ‘#1’ isn’t already defined}{}%
3609     }%
3610     {%
3611         \csdef{@glsstyle@#1}{#2}%
3612     }%
3613 }

```

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`

```
3614 \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`

```
3615 \ifcsundef{hyperlink}%
3616 {%
3617   \def\glshypernumber#1{#1}%
3618 }%
3619 {%
3620   \def\glshypernumber#1{\@glshypernumber#1\nohyperpage{}}\@nil}
3621 }
```

`\@glshypernumber` This code was provided by Heiko Oberdiek to allow material to be attached to the location.

```
3622 \def\@glshypernumber#1\nohyperpage#2#3\@nil{%
3623   \ifx\#1\%
3624     \else
3625       \@delimR#1\delimR\delimR\%
3626     \fi
3627   \ifx\#2\%
3628     \else
3629       #2%
3630     \fi
3631   \ifx\#3\%
3632     \else
3633       \@glshypernumber#3\@nil
3634     \fi
3635 }
```

`\@delimR` displays a range of numbers for the counter whose name is given by `\@gls@counter` (which must be set prior to using `\glshypernumber`).

`\@delimR`

```
3636 \def\@delimR#1\delimR #2\delimR #3\{%
3637 \ifx\#2\%
3638   \@delimN{#1}%
3639 \else
3640   \@gls@numberlink{#1}\delimR\@gls@numberlink{#2}%
3641 \fi}
```

`\@delimN` displays a list of individual numbers, instead of a range:

`\@delimN`

```

3642 \def\@delimN#1{\@delimN#1\delimN \delimN\}
3643 \def\@delimN#1\delimN #2\delimN#3\{\{
3644 \ifx\#3\}%
3645   \@gls@numberlink{#1}%
3646 \else
3647   \@gls@numberlink{#1}\delimN\@gls@numberlink{#2}%
3648 \fi
3649 }
```

The following code is modified from hyperref's `\HyInd@pagelink` where the name of the counter being used is given by `\@gls@counter`.

```

3650 \def\@gls@numberlink#1{%
3651 \begingroup
3652 \toks@={}%
3653 \@gls@removespaces#1 \@nil
3654 \endgroup}

3655 \def\@gls@removespaces#1 #2\@nil{%
3656 \toks@=\expandafter{\the\toks@#1}%
3657 \ifx\#2\}%
3658   \edef\x{\the\toks@}%
3659   \ifx\x\empty
3660     \else

3661     \hyperlink{\glsentrycounter\@glo@counterprefix\the\toks@}%
3662               {\the\toks@}%
3663   \fi
3664 \else
3665   \@gls@ReturnAfterFi{%
3666     \@gls@removespaces#2\@nil
3667   }%
3668 \fi
3669 }
3670 \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`

```

3671 \newcommand*\hyperrm[1]{\textrm{\glshypernumber{#1}}}
```

`\hypersf`

```

3672 \newcommand*\hypersf[1]{\textsf{\glshypernumber{#1}}}
```

`\hypertt`

```

3673 \newcommand*\hypertt[1]{\texttt{\glshypernumber{#1}}}
```

```

\hyperbf
3674 \newcommand*{\hyperbf}[1]{\textbf{\glshypernumber{#1}}}

\hypermd
3675 \newcommand*{\hypermd}[1]{\textmd{\glshypernumber{#1}}}

\hyperit
3676 \newcommand*{\hyperit}[1]{\textit{\glshypernumber{#1}}}

\hypersl
3677 \newcommand*{\hypersl}[1]{\textsl{\glshypernumber{#1}}}

\hyperup
3678 \newcommand*{\hyperup}[1]{\textup{\glshypernumber{#1}}}

\hypersc
3679 \newcommand*{\hypersc}[1]{\textsc{\glshypernumber{#1}}}

\hyperemph
3680 \newcommand*{\hyperemph}[1]{\emph{\glshypernumber{#1}}}

```

1.16 Acronyms

If the acronym package option is used, a new glossary called acronym is created

```

3681 \ifglsacronym
3682   \newglossary[alg]{acronym}{acr}{acn}{\acronymname}
      and \acronymtype is set to the name of this new glossary.
3683   \renewcommand*{\acronymtype}{acronym}
3684 \fi

```

```
\oldacronym \oldacronym[⟨label⟩]{⟨abbrv⟩}{⟨long⟩}{⟨key-val list⟩}
```

This emulates the way the old package defined acronyms. It is equivalent to `\newacronym[⟨key-val list⟩]{⟨label⟩}{⟨abbrv⟩}{⟨long⟩}` and it additionally defines the command `\⟨label⟩` which is equivalent to `\gls{⟨label⟩}` (thus `⟨label⟩` must only contain alphabetical characters). If `⟨label⟩` is omitted, `⟨abbrv⟩` is used. This only emulates the syntax of the old package. The way the acronyms appear in the list of acronyms is determined by the definition of `\newacronym` and the glossary style.

Note that `\⟨label⟩` can't have an optional argument if the package is loaded. If hasn't been loaded then you can do `\⟨label⟩[⟨insert⟩]` but you can't do `\⟨label⟩[⟨key-val list⟩]`. For example if you define the acronym `svm`, then you can do `\svm['s]` but you can't do `\svm[format=textbf]`. If the package is loaded, `\svm['s]` will appear as `svm ['s]` which is unlikely to be the desired

result. In this case, you will need to use `\gls` explicitly, e.g. `\gls{svm}[’s]`.
 Note that it is up to the user to load if desired.

```

3685 \newcommand{\oldacronym}[4][\gls@label]{%
3686   \def\gls@label{#2}%
3687   \newacronym[#4]{#1}{#2}{#3}%
3688   \ifcsundef{xspace}%
3689   {%
3690     \expandafter\edef\csname#1\endcsname{%
3691       \noexpand\@ifstar{\noexpand\Gls{#1}}{\noexpand\gls{#1}}%
3692     }%
3693   }%
3694   {%
3695     \expandafter\edef\csname#1\endcsname{%
3696       \noexpand\@ifstar{\noexpand\Gls{#1}\noexpand\xspace}{%
3697         \noexpand\gls{#1}\noexpand\xspace}%
3698     }%
3699   }%
3700 }
```

`\newacronym[⟨key-val list⟩]{⟨label⟩}{⟨abbrev⟩}{⟨long⟩}`

This is a quick way of defining acronyms, all it does is call `\newglossaryentry` with the appropriate values. It sets the glossary type to `\acronymtype` which will be `acronym` if the package option `acronym` has been used, otherwise it will be the default glossary. Since `\newacronym` merely calls `\newglossaryentry`, the acronym is treated like any other glossary entry.

If you prefer a different format, you can redefine `\newacronym` as required. The optional argument can be used to override any of the settings.

This is just a stub. It’s redefined by commands like `\SetDefaultAcronymStyle`.

`\newacronym`

```

3701 \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 need to cancel it out.

```

3702 \newcommand*{\acrpluralsuffix}{\glspluralsuffix}
```

The following are defined for compatibility with version 2.07 and earlier.

`\glsshortkey`

```

3703 \newcommand*{\glsshortkey}{short}
```

```

\glsshortpluralkey
3704 \newcommand*{\glsshortpluralkey}{shortplural}

\glslongkey
3705 \newcommand*{\glslongkey}{long}

\glslongpluralkey
3706 \newcommand*{\glslongpluralkey}{longplural}

\acrfull  Full form of the acronym.
3707 \newrobustcmd*{\acrfull}{%
3708   \@ifstar\s@acrfull\ns@acrfull
3709 }

3710 \newcommand*\s@acrfull[2][{}]{%
3711   \new@ifnextchar[{\@acrfull{hyper=false,#1}{#2}}}%
3712   {\@acrfull{hyper=false,#1}{#2}[]}%
3713 }
3714 \newcommand*\ns@acrfull[2][{}]{%
3715   \new@ifnextchar[{\@acrfull{#1}{#2}}}%
3716   {\@acrfull{#1}{#2}[]}%
3717 }

  Low-level macro:
3718 \def\@acrfull#1#2[#3]{%
3719   \acrlinkfullformat{\@acrlong}{\@acrshort}{#1}{#2}{#3}%
3720 }

\acrlinkfullformat  Format for full links like \acrfull.  Syntax: \acrlinkfullformat{<long
cs>}{<short cs>}{<options>}{<label>}{<insert>}
3721 \newcommand{\acrlinkfullformat}[5]{%
3722   \acrfullformat{#1{#3}{#4}[#5]}{#2{#3}{#4}[]}%
3723 }

\acrfullformat  Default full form is <long> (<short>).
3724 \newcommand{\acrfullformat}[2]{#1\space(#2)}

  Default format for full acronym

\Acrfull
3725 \newrobustcmd*{\Acrfull}{%
3726   \@ifstar\s@Acrfull\ns@Acrfull
3727 }

3728 \newcommand*\s@Acrfull[2][{}]{%
3729   \new@ifnextchar[{\@Acrfull{hyper=false,#1}{#2}}}%
3730   {\@Acrfull{hyper=false,#1}{#2}[]}%
3731 }
3732 \newcommand*\ns@Acrfull[2][{}]{%

```

```

3733 \new@ifnextchar[{\@Acrfull{#1}{#2}}%
3734         {\@Acrfull{#1}{#2} []}%
3735 }

```

Low-level macro:

```

3736 \def\@Acrfull#1#2[#3]{%
3737 \acrlinkfullformat{\@Acrlong}{\@acrshort}{#1}{#2}{#3}%
3738 }

```

\ACRfull

```

3739 \newrobustcmd*{\ACRfull}{%
3740 \@ifstar\s@ACRfull\ns@ACRfull
3741 }

3742 \newcommand*\s@ACRfull[2] []{%
3743 \new@ifnextchar[{\@ACRfull{hyper=false,#1}{#2}}%
3744         {\@ACRfull{hyper=false,#1}{#2} []}%
3745 }

3746 \newcommand*\ns@ACRfull[2] []{%
3747 \new@ifnextchar[{\@ACRfull{#1}{#2}}%
3748         {\@ACRfull{#1}{#2} []}%
3749 }

```

Low-level macro:

```

3750 \def\@ACRfull#1#2[#3]{%
3751 \acrlinkfullformat{\@ACRlong}{\@ACRshort}{#1}{#2}{#3}%
3752 }

```

Plural:

\acrfullpl

```

3753 \newrobustcmd*{\acrfullpl}{%
3754 \@ifstar\s@acrfullpl\ns@acrfullpl
3755 }

3756 \newcommand*\s@acrfullpl[2] []{%
3757 \new@ifnextchar[{\@acrfullpl{hyper=false,#1}{#2}}%
3758         {\@acrfullpl{hyper=false,#1}{#2} []}%
3759 }

3760 \newcommand*\ns@acrfullpl[2] []{%
3761 \new@ifnextchar[{\@acrfullpl{#1}{#2}}%
3762         {\@acrfullpl{#1}{#2} []}%
3763 }

```

Low-level macro:

```

3764 \def\@acrfullpl#1#2[#3]{%
3765 \acrlinkfullformat{\@acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%
3766 }

```

`\Acrfullpl`

```
3767 \newrobustcmd*{\Acrfullpl}{%
3768   \@ifstar\s@Acrfullpl\ns@Acrfullpl
3769 }

3770 \newcommand*\s@Acrfullpl[2][]{%
3771   \new@ifnextchar[{\@Acrfullpl{hyper=false,#1}{#2}}%
3772     {\@Acrfullpl{hyper=false,#1}{#2}[]}%
3773 }
3774 \newcommand*\ns@Acrfullpl[2][]{%
3775   \new@ifnextchar[{\@Acrfullpl{#1}{#2}}%
3776     {\@Acrfullpl{#1}{#2}[]}%
3777 }
```

Low-level macro:

```
3778 \def\@Acrfullpl#1#2[#3]{%
3779   \acrlinkfullformat{\@Acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%
3780 }
```

`\ACRfullpl`

```
3781 \newrobustcmd*{\ACRfullpl}{%
3782   \@ifstar\s@ACRfullpl\ns@ACRfullpl
3783 }

3784 \newcommand*\s@ACRfullpl[2][]{%
3785   \new@ifnextchar[{\@ACRfullpl{hyper=false,#1}{#2}}%
3786     {\@ACRfullpl{hyper=false,#1}{#2}[]}%
3787 }
3788 \newcommand*\ns@ACRfullpl[2][]{%
3789   \new@ifnextchar[{\@ACRfullpl{#1}{#2}}%
3790     {\@ACRfullpl{#1}{#2}[]}%
3791 }
```

Low-level macro:

```
3792 \def\@ACRfullpl#1#2[#3]{%
3793   \acrlinkfullformat{\@ACRlongpl}{\@ACRshortpl}{#1}{#2}{#3}%
3794 }
```

1.17 Predefined acronym styles

`\acronymfont` This is only used with the additional acronym styles:

```
3795 \newcommand{\acronymfont}[1]{#1}
```

`\firstacronymfont` This is only used with the additional acronym styles:

```
3796 \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.

```
3797 \newcommand*\acrnameformat[2]{\acronymfont{#1}}
```


Define some tokens used by \newacronym:

\glskeylisttok

3798 \newtoks\glskeylisttok

\glslabeltok

3799 \newtoks\glslabeltok

\glsshorttok

3800 \newtoks\glsshorttok

\glslongtok

3801 \newtoks\glslongtok

\newacronymhook Provide a hook for \newacronym:

3802 \newcommand*{\newacronymhook}{}

AcronymDisplayStyle Sets the default acronym display style for given glossary.

3803 \newcommand*{\SetDefaultAcronymDisplayStyle}[1]{%

3804 \defglsdisplay[#1]{##1##4}%

3805 \defglsdisplayfirst[#1]{##1##4}%

3806 }

DefaultNewAcronymDef Sets up the acronym definition for the default style. The information is provided by the tokens \glslabeltok, \glsshorttok, \glslongtok and \glskeylisttok.

3807 \newcommand*{\DefaultNewAcronymDef}{%

3808 \edef\@do@newglossaryentry{%

3809 \noexpand\newglossaryentry{\the\glslabeltok}%

3810 {%

3811 type=\acronymtype,%

3812 name={\the\glsshorttok},%

3813 sort={\the\glsshorttok},%

3814 text={\the\glsshorttok},%

3815 first={\acrfullformat{\the\glslongtok}{\the\glsshorttok}},%

3816 plural={\the\glsshorttok\noexpand\acrpluralsuffix},%

3817 firstplural={\acrfullformat{\noexpand\@glo@longpl}%

3818 {\noexpand\@glo@shortpl}},%

3819 short={\the\glsshorttok},%

3820 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%

3821 long={\the\glslongtok},%

3822 longplural={\the\glslongtok\noexpand\acrpluralsuffix},%

3823 description={\the\glslongtok},%

3824 descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%

Remaining options specified by the user:

3825 \the\glskeylisttok

3826 }%

```

3827 }%
3828 \@do@newglossaryentry
3829 }

```

DefaultAcronymStyle Set up the default acronym style:

```

3830 \newcommand*\SetDefaultAcronymStyle{%
    Set the display style:
3831   \@for\@gls@type:=\@glsacronymlists\do{%
3832     \SetDefaultAcronymDisplayStyle{\@gls@type}%
3833   }%

```

Set up the definition of \newacronym:

```

3834 \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).
3835   \ifx\@glsacronymlists\@empty
3836     \def\@glo@type{\acronymtype}%
3837     \setkeys{glossentry}{##1}%
3838     \DeclareAcronymList{\@glo@type}%
3839     \SetDefaultAcronymDisplayStyle{\@glo@type}%
3840   \fi
3841   \glskeylisttok{##1}%
3842   \glslabeltok{##2}%
3843   \glsshorttok{##3}%
3844   \glslongtok{##4}%
3845   \newacronymhook
3846   \DefaultNewAcronymDef
3847 }%
3848 \renewcommand*\acrpluralsuffix{\glspluralsuffix}%
3849 }

```

\acrfootnote Used by the footnote acronym styles.

```

3850 \newcommand*\acrfootnote}[3]{\acrlinkfootnote{#1}{#2}{#3}}

```

\acrlinkfootnote

```

3851 \newcommand*\acrlinkfootnote}[3]{%
3852   \footnote{\glslink[#1]{#2}{#3}}%
3853 }

```

\acrnoflinkfootnote

```

3854 \newcommand*\acrnoflinkfootnote}[3]{%
3855   \footnote{#3}%
3856 }

```

AcronymDisplayStyle Sets the acronym display style for given glossary for the description and footnote combination.

```

3857 \newcommand*{\SetDescriptionFootnoteAcronymDisplayStyle}[1]{%
3858   \defglsdisplayfirst[#1]{%
3859     \firstacronymfont{##1}##4%
3860     \expandafter\protect\expandafter\acrfootnote\expandafter
3861       {\@gls@link@opts}{\@gls@link@label}{##3}%
3862   }%
3863   \defglsdisplay[#1]{\acronymfont{##1}##4}%
3864 }

```

otnoteNewAcronymDef

```

3865 \newcommand*{\DescriptionFootnoteNewAcronymDef}{%
3866   \edef\@do@newglossaryentry{%
3867     \noexpand\newglossaryentry{\the\glslabeltok}%
3868     {%
3869       type=\acronymtype,%
3870       name={\noexpand\acronymfont{\the\glsshorttok}},%
3871       sort={\the\glsshorttok},%
3872       text={\the\glsshorttok},%
3873       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3874       short={\the\glsshorttok},%
3875       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3876       long={\the\glslongtok},%
3877       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3878       symbol={\the\glslongtok},%
3879       symbolplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3880       \the\glskeylisttok
3881     }%
3882   }%
3883   \@do@newglossaryentry
3884 }

```

ootnoteAcronymStyle If a description and footnote are both required, store the long form in the symbol key. Store the short form in text key. Note that since the long form is stored in the symbol key, if you want the long form to appear in the list of acronyms, you need to use a glossary style that displays the symbol key.

```

3885 \newcommand*{\SetDescriptionFootnoteAcronymStyle}{%
3886   \renewcommand{\newacronym}[4][\]{%
3887     \ifx\@glsacronymlists\@empty
3888       \def\@glo@type{\acronymtype}%
3889       \setkeys{glossentry}{##1}%
3890       \DeclareAcronymList{\@glo@type}%
3891       \SetDescriptionFootnoteAcronymDisplayStyle{\@glo@type}%
3892     \fi
3893     \glskeylisttok{##1}%
3894     \glslabeltok{##2}%
3895     \glsshorttok{##3}%
3896     \glslongtok{##4}%
3897     \newacronymhook
3898     \DescriptionFootnoteNewAcronymDef

```

3899 }%

If footnote package option is specified, set the first use to append the long form (stored in symbol) as a footnote.

```
3900 \@for\@gls@type:=\@glsacronymlists\do{%
3901     \SetDescriptionFootnoteAcronymDisplayStyle{\@gls@type}%
3902 }%
```

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.

```
3903 \ifglsacrsmallcaps
3904     \renewcommand*{\acronymfont}[1]{\textsc{##1}}%
3905     \renewcommand*{\acrpluralsuffix}{%
3906         \textup{\glspluralsuffix}}%
3907 \else
3908     \ifglsacrsmaller
3909         \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%
3910     \fi
3911 \fi
```

Check for package option clash

```
3912 \ifglsacrdua
3913     \PackageError{glossaries}{Option clash: ‘footnote’ and ‘dua’
3914         can’t both be set}{}%
3915 \fi
3916 }%
```

`AcronymDisplayStyle` Sets the acronym display style for given glossary with description and dua combination.

```
3917 \newcommand*{\SetDescriptionDUAAcronymDisplayStyle}[1]{%
3918     \defglsdisplay[#1]{##1##4}%
3919     \defglsdisplayfirst[#1]{##1##4}%
3920 }
```

`ionDUANewAcronymDef`

```
3921 \newcommand*{\DescriptionDUANewAcronymDef}{%
3922     \edef\@do@newglossaryentry{%
3923         \noexpand\newglossaryentry{\the\glslabeltok}%
3924         {%
3925             type=\acronymtype,%
3926             name={\the\glslongtok},%
3927             sort={\the\glslongtok},%
3928             text={\the\glslongtok},%
3929             plural={\the\glslongtok\noexpand\acrpluralsuffix},%
3930             short={\the\glsshorttok},%
3931             shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3932             long={\the\glslongtok},%
3933             longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3934             symbol={\the\glsshorttok},%
```

```

3935     symbolplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3936     \the\glskeylisttok
3937   }%
3938 }%
3939 \@do@newglossaryentry
3940 }

```

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.

```

3941 \newcommand*{\SetDescriptionDUAAcronymStyle}{%
3942   \ifglsacrsmallcaps
3943     \PackageError{glossaries}{Option clash: 'smallcaps' and 'dua'
3944       can't both be set}{}%
3945   \else
3946     \ifglsacrsmaller
3947       \PackageError{glossaries}{Option clash: 'smaller' and 'dua'
3948         can't both be set}{}%
3949     \fi
3950   \fi
3951   \renewcommand{\newacronym}[4][{}]{%
3952     \ifx\@glsacronymlists\@empty
3953       \def\@glo@type{\acronymtype}%
3954       \setkeys{glossentry}{##1}%
3955       \DeclareAcronymList{\@glo@type}%
3956       \SetDescriptionDUAAcronymDisplayStyle{\@glo@type}%
3957     \fi
3958     \glskeylisttok{##1}%
3959     \glslabeltok{##2}%
3960     \glsshorttok{##3}%
3961     \glslongtok{##4}%
3962     \newacronymhook
3963     \DescriptionDUANewAcronymDef
3964   }%

```

Set display.

```

3965   \@for\@gls@type:=\@glsacronymlists\do{%
3966     \SetDescriptionDUAAcronymDisplayStyle{\@gls@type}%
3967   }%
3968 }%

```

AcronymDisplayStyle Sets the acronym display style for given glossary using the description setting (but not footnote or dua).

```

3969 \newcommand*{\SetDescriptionAcronymDisplayStyle}[1]{%
3970   \defglsdisplayfirst[#1]{%
3971     ##1##4 (\firstacronymfont{##3})}%
3972   \defglsdisplay[#1]{\acronymfont{##1}##4}%
3973 }

```

ptionNewAcronymDef

```

3974 \newcommand*{\DescriptionNewAcronymDef}{%
3975   \edef\@do@newglossaryentry{%
3976     \noexpand\newglossaryentry{\the\glslabeltok}%
3977     {%
3978       type=\acronymtype,%
3979       name={\noexpand
3980         \acrnameformat{\the\glsshorttok}{\the\glslongtok}},%
3981       sort={\the\glsshorttok},%
3982       first={\the\glslongtok},%
3983       firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3984       text={\the\glsshorttok},%
3985       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3986       short={\the\glsshorttok},%
3987       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3988       long={\the\glslongtok},%
3989       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3990       symbol={\noexpand\@glo@text},%
3991       symbolplural={\noexpand\@glo@plural},%
3992       \the\glskeylisttok}%
3993   }%
3994   \@do@newglossaryentry
3995 }

```

riptionAcronymStyle

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.

```

3996 \newcommand*{\SetDescriptionAcronymStyle}{%
3997   \renewcommand{\newacronym}[4][\]{%
3998     \ifx\@glsacronymlists\@empty
3999       \def\@glo@type{\acronymtype}%
4000       \setkeys{glossentry}{##1}%
4001       \DeclareAcronymList{\@glo@type}%
4002       \SetDescriptionAcronymDisplayStyle{\@glo@type}%
4003     \fi
4004     \glskeylisttok{##1}%
4005     \glslabeltok{##2}%
4006     \glsshorttok{##3}%
4007     \glslongtok{##4}%
4008     \newacronymhook
4009     \DescriptionNewAcronymDef
4010   }%

```

Set display.

```

4011 \for\@gls@type:=\@glsacronymlists\do{%
4012   \SetDescriptionAcronymDisplayStyle{\@gls@type}%
4013 }%

```

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.

```

4014 \ifglsmallcaps
4015   \renewcommand{\acronymfont}[1]{\textsc{##1}}
4016   \renewcommand*{\acrpluralsuffix}{%
4017     \textup{\glpluralsuffix}}%
4018   \else
4019     \ifglsmaller
4020       \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%
4021     \fi
4022   \fi
4023 }%
```

AcronymDisplayStyle Sets the acronym display style for given glossary with footnote setting (but not description or dua).

```

4024 \newcommand*\SetFootnoteAcronymDisplayStyle[1]{%
4025   \defgldisplayfirst[#1]{%
4026     \firstacronymfont{##1}##4%
4027     \expandafter\protect\expandafter\acrfootnote\expandafter
4028       {\@gls@link@opts}{\@gls@link@label}{##2}%
4029   }%
4030   \defgldisplay[#1]{\acronymfont{##1}##4}%
4031 }
```

FootnoteNewAcronymDef

```

4032 \newcommand*\FootnoteNewAcronymDef{%
4033   \edef\@do@newglossaryentry{%
4034     \noexpand\newglossaryentry{\the\glslabeltok}%
4035     {%
4036       type=\acronymtype,%
4037       name={\noexpand\acronymfont{\the\glsshorttok}},%
4038       sort={\the\glsshorttok},%
4039       text={\the\glsshorttok},%
4040       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4041       short={\the\glsshorttok},%
4042       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4043       long={\the\glslongtok},%
4044       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4045       description={\the\glslongtok},%
4046       descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4047       \the\glskeylisttok
4048     }%
4049   }%
4050   \@do@newglossaryentry
4051 }
```

FootnoteAcronymStyle If footnote package option is specified, set the first use to append the long form

(stored in description) as a footnote. Use the description key to store the long form.

```

4052 \newcommand*\SetFootnoteAcronymStyle{%
4053   \renewcommand{\newacronym}[4][\]{%
4054     \ifx\@glsacronymlists\@empty
4055       \def\@glo@type{\acronymtype}%
4056       \setkeys{glossentry}{##1}%
4057       \DeclareAcronymList{\@glo@type}%
4058       \SetFootnoteAcronymDisplayStyle{\@glo@type}%
4059     \fi
4060     \glskeylisttok{##1}%
4061     \glslabeltok{##2}%
4062     \glsshorttok{##3}%
4063     \glslongtok{##4}%
4064     \newacronymhook
4065     \FootnoteNewAcronymDef
4066   }%

```

Set display

```

4067   \@for\@gls@type:=\@glsacronymlists\do{%
4068     \SetFootnoteAcronymDisplayStyle{\@gls@type}%
4069   }%

```

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.

```

4070   \ifglsacrsmallcaps
4071     \renewcommand*\acronymfont[1]{\textsc{##1}}%
4072     \renewcommand*\acrpluralsuffix{%
4073       \textup{\glspluralsuffix}}%
4074   \else
4075     \ifglsacrsmaller
4076       \renewcommand*\acronymfont[1]{\textsmaller{##1}}%
4077     \fi
4078   \fi

```

Check for option clash

```

4079   \ifglsacrdua
4080     \PackageError{glossaries}{Option clash: ‘footnote’ and ‘dua’
4081       can’t both be set}{}%
4082   \fi
4083 }%

```

AcronymDisplayStyle Sets the acronym display style for given glossary where neither footnote nor description is required, but smallcaps or smaller specified.

```

4084 \newcommand*\SetSmallAcronymDisplayStyle[1]{%
4085   \defglsdisplayfirst[#1]{##1##4 (\firstacronymfont{##3})}
4086   \defglsdisplay[#1]{\acronymfont{##1}##4}%
4087 }

```


\SmallNewAcronymDef

```

4088 \newcommand*{\SmallNewAcronymDef}{%
4089   \edef\@do@newglossaryentry{%
4090     \noexpand\newglossaryentry{\the\glslabeltok}%
4091     {%
4092       type=\acronymtype,%
4093       name={\noexpand\acronymfont{\the\glsshorttok}},%
4094       sort={\the\glsshorttok},%
4095       text={\noexpand\@glo@symbol},%
4096       %plural={\noexpand\@glo@symbolplural},%
4097       plural={\noexpand\@glo@shortpl},%
4098       first={\the\glslongtok},%
4099       %firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4100       firstplural={\noexpand\@glo@longpl},%
4101       short={\the\glsshorttok},%
4102       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4103       long={\the\glslongtok},%
4104       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4105       description={\noexpand\@glo@first},%
4106       descriptionplural={\noexpand\@glo@firstplural},%
4107       symbol={\the\glsshorttok},%
4108       %symbolplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4109       symbolplural={\noexpand\@glo@shortpl},%
4110       \the\glskeylisttok
4111     }%
4112   }%
4113   \@do@newglossaryentry
4114 }

```

etSmallAcronymStyle 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.

```

4115 \newcommand*{\SetSmallAcronymStyle}{%
4116   \renewcommand{\newacronym}[4][\]{%
4117     \ifx\@glsacronymlists\@empty
4118       \def\@glo@type{\acronymtype}%
4119       \setkeys{glossentry}{##1}%
4120       \DeclareAcronymList{\@glo@type}%
4121       \SetSmallAcronymDisplayStyle{\@glo@type}%
4122     \fi
4123     \glskeylisttok{##1}%
4124     \glslabeltok{##2}%
4125     \glsshorttok{##3}%
4126     \glslongtok{##4}%
4127     \newacronymhook
4128     \SmallNewAcronymDef
4129   }%

```

Change the display since first only contains long form.

```

4130   \@for\@gls@type:=\@glsacronymlists\do{%

```

```

4131 \SetSmallAcronymDisplayStyle{\@gls@type}%
4132 }%

```

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.

```

4133 \ifglssacrsmallcaps
4134 \renewcommand*{\acronymfont}[1]{\textsc{##1}}
4135 \renewcommand*{\acrpluralsuffix}{%
4136 \textup{\glspluralsuffix}}%
4137 \else
4138 \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}
4139 \fi

```

check for option clash

```

4140 \ifglssacrdua
4141 \ifglssacrsmallcaps
4142 \PackageError{glossaries}{Option clash: 'smallcaps' and 'dua'
4143 can't both be set}{}%
4144 \else
4145 \PackageError{glossaries}{Option clash: 'smaller' and 'dua'
4146 can't both be set}{}%
4147 \fi
4148 \fi
4149 }%

```

\SetDUADisplayStyle Sets the acronym display style for given glossary with dua setting.

```

4150 \newcommand*{\SetDUADisplayStyle}[1]{%
4151 \defglssdisplay[##1]{##1##4}%
4152 \defglssdisplayfirst[##1]{##1##4}%
4153 }

```

\DUANewAcronymDef

```

4154 \newcommand*{\DUANewAcronymDef}{%
4155 \edef\@do@newglossaryentry{%
4156 \noexpand\newglossaryentry{\the\glslabeltok}%
4157 {%
4158 type=\acronymtype,%
4159 name={\the\glsshorttok},%
4160 text={\the\glslongtok},%
4161 plural={\the\glslongtok\noexpand\acrpluralsuffix},%
4162 short={\the\glsshorttok},%
4163 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4164 long={\the\glslongtok},%
4165 longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4166 description={\the\glslongtok},%
4167 symbol={\the\glsshorttok},%
4168 symbolplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4169 \the\glskeylisttok
4170 }%

```

```

4171 }%
4172 \@do@newglossaryentry
4173 }

```

`\SetDUASStyle` Always expand acronyms.

```

4174 \newcommand*{\SetDUASStyle}{%
4175   \renewcommand{\newacronym}[4][]{%
4176     \ifx\@glsacronymlists\@empty
4177       \def\@glo@type{\acronymtype}%
4178       \setkeys{glossentry}{##1}%
4179       \DeclareAcronymList{\@glo@type}%
4180       \SetDUADisplayStyle{\@glo@type}%
4181     \fi
4182     \glskeylisttok{##1}%
4183     \glslabeltok{##2}%
4184     \glsshorttok{##3}%
4185     \glslongtok{##4}%
4186     \newacronymhook
4187     \DUANewAcronymDef
4188   }%

```

Set the display

```

4189   \@for\@gls@type:=\@glsacronymlists\do{%
4190     \SetDUADisplayStyle{\@gls@type}%
4191   }%
4192 }

```

`\SetAcronymStyle`

```

4193 \newcommand*{\SetAcronymStyle}{%
4194   \SetDefaultAcronymStyle
4195   \ifglsacrdescription
4196     \ifglsacrfootnote
4197       \SetDescriptionFootnoteAcronymStyle
4198     \else
4199       \ifglsacrdua
4200         \SetDescriptionDUAAcronymStyle
4201       \else
4202         \SetDescriptionAcronymStyle
4203       \fi
4204     \fi
4205   \else
4206     \ifglsacrfootnote
4207       \SetFootnoteAcronymStyle
4208     \else
4209       \ifthenelse{\boolean{glsacrsmalldcaps}}{OR
4210         \boolean{glsacrsmaller}}{%
4211         {%
4212           \SetSmallAcronymStyle
4213         }%
4214       {%

```

```

4215         \ifglsacrdua
4216         \SetDUASstyle
4217     \fi
4218 }%
4219 \fi
4220 \fi
4221 }

```

Set the acronym style according to the package options

```

4222 \SetAcronymStyle

```

Allow user to define their own custom acronyms. (For compatibility with versions before v3.0, the short form is stored in the user1 key, the plural short form is stored in the user2 key, the long form is stored in the user3 key and the plural long form is stored in the user4 key.) Defaults to displaying only the acronym with the long form as the description.

CustomDisplayStyle Sets the acronym display style.

```

4223 \newcommand*\SetCustomDisplayStyle[1]{%
4224     \defglsdisplay[#1]{##1##4}%
4225     \defglsdisplayfirst[#1]{##1##4}%
4226 }

```

CustomAcronymFields

```

4227 \newcommand*\CustomAcronymFields{%
4228     name={\the\glsshorttok},%
4229     description={\the\glslongtok},%
4230     first={\noexpand\acrfullformat{\the\glslongtok}{\the\glsshorttok}},%
4231     firstplural={\noexpand\acrfullformat
4232         {\the\glslongtok\noexpand\acrpluralsuffix}{\the\glsshorttok}}%
4233     text={\the\glsshorttok},%
4234     plural={\the\glsshorttok\noexpand\acrpluralsuffix}%
4235 }

```

CustomNewAcronymDef

```

4236 \newcommand*\CustomNewAcronymDef{%
4237     \protected@edef\@do@newglossaryentry{%
4238         \noexpand\newglossaryentry{\the\glslabeltok}%
4239         {%
4240             type=\acronymtype,%
4241             short={\the\glsshorttok},%
4242             shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4243             long={\the\glslongtok},%
4244             longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4245             user1={\the\glsshorttok},%
4246             user2={\the\glsshorttok\noexpand\acrpluralsuffix},%
4247             user3={\the\glslongtok},%
4248             user4={\the\glslongtok\noexpand\acrpluralsuffix},%
4249             \CustomAcronymFields,%

```

```

4250     \the\glskeylisttok
4251   }%
4252 }%
4253 \do@newglossaryentry
4254 }

```

\SetCustomStyle

```

4255 \newcommand*{\SetCustomStyle}{%
4256   \renewcommand{\newacronym}[4][[]]{%
4257     \ifx\@glsacronymlists\@empty
4258       \def\@glo@type{\acronymtype}%
4259       \setkeys{glossentry}{##1}%
4260       \DeclareAcronymList{\@glo@type}%
4261       \SetCustomDisplayStyle{\@glo@type}%
4262     \fi
4263     \glskeylisttok{##1}%
4264     \glslabeltok{##2}%
4265     \glsshorttok{##3}%
4266     \gslongtok{##4}%
4267     \newacronymhook
4268     \CustomNewAcronymDef
4269   }%

```

Set the display

```

4270   \@for\@gls@type:=\@glsacronymlists\do{%
4271     \SetCustomDisplayStyle{\@gls@type}%
4272   }%
4273 }

```

fineAcronymSynonyms

```

4274 \newcommand*{\DefineAcronymSynonyms}{%

```

Short form

\acs

```

4275   \let\acs\acrshort

```

First letter uppercase short form

\Acs

```

4276   \let\Acs\Acrshort

```

Plural short form

\acsp

```

4277   \let\acsp\acrshortpl

```

First letter uppercase plural short form

\Acsp

```

4278   \let\Acsp\Acrshortpl

```

Long form

\acl

4279 \let\acl\acrlong

Plural long form

\aclp

4280 \let\aclp\acrlongpl

First letter upper case long form

\Acl

4281 \let\Acl\Acrlong

First letter upper case plural long form

\Aclp

4282 \let\Aclp\Acrlongpl

Full form

\acf

4283 \let\acf\acrfull

Plural full form

\acfp

4284 \let\acfp\acrfullpl

First letter upper case full form

\Acf

4285 \let\Acf\Acrfull

First letter upper case plural full form

\Acfp

4286 \let\Acfp\Acrfullpl

Standard form

\ac

4287 \let\ac\gls

First upper case standard form

\Ac

4288 \let\Ac\Gls

Standard plural form

\acp

4289 \let\acp\glspl

Standard first letter upper case plural form

`\Acp`

```
4290 \let\Acp\Glspl
```

```
4291 }
```

Define synonyms if required

```
4292 \ifglsacrshortcuts
```

```
4293 \DefineAcronymSynonyms
```

```
4294 \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.

```
4295 \RequirePackage{glossary-hypernav}
```

The styles that use list-like environments. These are not loaded if the `nolist` option is used:

```
4296 \@gls@loadlist
```

The styles that use the `longtable` environment. These are not loaded if the `no-long package` option is used.

```
4297 \@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.

```
4298 \@gls@loadsuper
```

The tree-like styles. These are not loaded if the `notree` package option is used.

```
4299 \@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.

```
4300 \ifx\@glossary@default@style\relax
```

```
4301 \else
```

```
4302 \glossarystyle{\@glossary@default@style}
```

```
4303 \fi
```

1.19 Debugging Commands

`\showgloparent` `\showgloparent{<label>}`

```
4304 \newcommand*{\showgloparent}[1]{%
```

```
4305 \expandafter\show\csname glo@#1@parent\endcsname
```

```
4306 }
```

`\showglolevel` `\showglolevel{<label>}`

```
4307 \newcommand*{\showglolevel}[1]{%  
4308   \expandafter\show\csname glo@#1@level\endcsname  
4309 }
```

`\showglotext` `\showglotext{<label>}`

```
4310 \newcommand*{\showglotext}[1]{%  
4311   \expandafter\show\csname glo@#1@text\endcsname  
4312 }
```

`\showgloplural` `\showgloplural{<label>}`

```
4313 \newcommand*{\showgloplural}[1]{%  
4314   \expandafter\show\csname glo@#1@plural\endcsname  
4315 }
```

`\showglofirst` `\showglofirst{<label>}`

```
4316 \newcommand*{\showglofirst}[1]{%  
4317   \expandafter\show\csname glo@#1@first\endcsname  
4318 }
```

`\showglofirstpl` `\showglofirstpl{<label>}`

```
4319 \newcommand*{\showglofirstpl}[1]{%  
4320   \expandafter\show\csname glo@#1@firstpl\endcsname  
4321 }
```

`\showglotype` `\showglotype{<label>}`

```
4322 \newcommand*{\showglotype}[1]{%  
4323   \expandafter\show\csname glo@#1@type\endcsname  
4324 }
```

`\showglocounter` `\showglocounter{<label>}`

```
4325 \newcommand*{\showglocounter}[1]{%  
4326   \expandafter\show\csname glo@#1@counter\endcsname  
4327 }
```


\showglouserii \showglouserii{<label>}

```
4328 \newcommand*{\showglouserii}[1]{%
4329   \expandafter\show\csname glo@#1@userii\endcsname
4330 }
```

\showglouseriii \showglouseriii{<label>}

```
4331 \newcommand*{\showglouseriii}[1]{%
4332   \expandafter\show\csname glo@#1@useriii\endcsname
4333 }
```

\showglouseriv \showglouseriv{<label>}

```
4334 \newcommand*{\showglouseriv}[1]{%
4335   \expandafter\show\csname glo@#1@useriv\endcsname
4336 }
```

\showglouserv \showglouserv{<label>}

```
4337 \newcommand*{\showglouserv}[1]{%
4338   \expandafter\show\csname glo@#1@userv\endcsname
4339 }
```

\showglouservi \showglouservi{<label>}

```
4340 \newcommand*{\showglouservi}[1]{%
4341   \expandafter\show\csname glo@#1@uservi\endcsname
4342 }
```

\showgloname \showgloname{<label>}

```
4343 \newcommand*{\showgloname}[1]{%
4344   \expandafter\show\csname glo@#1@name\endcsname
4345 }
```

\showgloname \showgloname{<label>}

```
4346 \newcommand*{\showgloname}[1]{%
4347   \expandafter\show\csname glo@#1@name\endcsname
4348 }
```

`\showglodesc` `\showglodesc{<label>}`

```
4349 \newcommand*{\showglodesc}[1]{%
4350   \expandafter\show\csname glo@#1@desc\endcsname
4351 }
```

`\showglodescplural` `\showglodescplural{<label>}`

```
4352 \newcommand*{\showglodescplural}[1]{%
4353   \expandafter\show\csname glo@#1@descplural\endcsname
4354 }
```

`\showglosort` `\showglosort{<label>}`

```
4355 \newcommand*{\showglosort}[1]{%
4356   \expandafter\show\csname glo@#1@sort\endcsname
4357 }
```

`\showglosymbol` `\showglosymbol{<label>}`

```
4358 \newcommand*{\showglosymbol}[1]{%
4359   \expandafter\show\csname glo@#1@symbol\endcsname
4360 }
```

`\showglosymbolplural` `\showglosymbolplural{<label>}`

```
4361 \newcommand*{\showglosymbolplural}[1]{%
4362   \expandafter\show\csname glo@#1@symbolplural\endcsname
4363 }
```

`\showgloindex` `\showgloindex{<label>}`

```
4364 \newcommand*{\showgloindex}[1]{%
4365   \expandafter\show\csname glo@#1@index\endcsname
4366 }
```

`\showgloflag` `\showgloflag{<label>}`

```
4367 \newcommand*{\showgloflag}[1]{%
4368   \expandafter\show\csname ifglo@#1@flag\endcsname
4369 }
```

`\showacronymlists`

`\showacronymlists`

Show list of glossaries that have been flagged as a list of acronyms.

```
4370 \newcommand*{\showacronymlists}{%  
4371   \show\@glsacronymlists  
4372 }
```

`\showglossaries`

`\showglossaries`

Show list of defined glossaries.

```
4373 \newcommand*{\showglossaries}{%  
4374   \show\@glo@types  
4375 }
```

`\showglossaryin`

`\showglossaryin{<glossary-label>}`

Show the ‘in’ extension for the given glossary.

```
4376 \newcommand*{\showglossaryin}[1]{%  
4377   \expandafter\show\csname @glo@type@#1@in\endcsname  
4378 }
```

`\showglossaryout`

`\showglossaryout{<glossary-label>}`

Show the ‘out’ extension for the given glossary.

```
4379 \newcommand*{\showglossaryout}[1]{%  
4380   \expandafter\show\csname @glo@type@#1@out\endcsname  
4381 }
```

`\showglossarytitle`

`\showglossarytitle{<glossary-label>}`

Show the title for the given glossary.

```
4382 \newcommand*{\showglossarytitle}[1]{%  
4383   \expandafter\show\csname @glo@type@#1@title\endcsname  
4384 }
```

`\showglossarycounter`

`\showglossarycounter{<glossary-label>}`

Show the counter for the given glossary.

```
4385 \newcommand*{\showglossarycounter}[1]{%  
4386   \expandafter\show\csname @glo@type@#1@counter\endcsname  
4387 }
```

```
\showglossaryentries \showglossaryentries{\glossary-label}
```

Show the list of entry labels for the given glossary.

```
4388 \newcommand*{\showglossaryentries}[1]{%
4389   \expandafter\show\csname glolist@#1\endcsname
4390 }
```

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.

```
4391 \csname ifglscpatible-2.07\endcsname
4392   \RequirePackage{glossaries-compatible-207}
4393 \fi
```

2 Mfirstuc Documented Code

```
4394 \NeedsTeXFormat{LaTeX2e}
4395 \ProvidesPackage{mfirstuc}[2012/05/21 v1.06 (NLCT)]
```

Requires etoolbox:

```
4396 \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`.

```
4397 \newif\if@glscs
4398 \newtoks\@glsmfirst
4399 \newtoks\@glsmrest
4400 \def\makefirstuc#1{%
4401   \def\gls@argi{#1}%
4402   \ifx\gls@argi\@empty
```

If the argument is empty, do nothing.

```

4403 \else

4404 \def\@gls@tmp{\ #1}%
4405 \@onelevel@sanitize\@gls@tmp
4406 \expandafter\@gls@checkcs\@gls@tmp\relax\relax
4407 \if@glscs
4408 \@gls@getbody #1{}\@nil
4409 \ifx\@gls@rest\@empty
4410 \glsmakefirstuc{#1}%
4411 \else
4412 \expandafter\@gls@split\@gls@rest\@nil
4413 \ifx\@gls@first\@empty
4414 \glsmakefirstuc{#1}%
4415 \else
4416 \expandafter\@glsmfirst\expandafter{\@gls@first}%
4417 \expandafter\@glsmrest\expandafter{\@gls@rest}%
4418 \edef\@gls@domfirstuc{\noexpand\@gls@body
4419 {\noexpand\glsmakefirstuc\the\@glsmfirst}%
4420 \the\@glsmrest}%
4421 \@gls@domfirstuc
4422 \fi
4423 \fi
4424 \else
4425 \glsmakefirstuc{#1}%
4426 \fi
4427 \fi
4428 }

```

Put first argument in \@gls@first and second argument in \@gls@rest:

```

4429 \def\@gls@split#1#2\@nil{%
4430 \def\@gls@first{#1}\def\@gls@rest{#2}%
4431 }

4432 \def\@gls@checkcs#1 #2#3\relax{%
4433 \def\@gls@argi{#1}\def\@gls@argii{#2}%
4434 \ifx\@gls@argi\@gls@argii
4435 \@glscstrue
4436 \else
4437 \@glscsfalse
4438 \fi
4439 }

```

Make first thing upper case:

```

4440 \def\@gls@makefirstuc#1{\MakeUppercase #1}

```

\glsmakefirstuc Provide a user command to make it easier to customise.

```

4441 \newcommand*\glsmakefirstuc}[1]{\@gls@makefirstuc{#1}}

```

Get the first grouped argument and stores in \@gls@body.

```
4442 \def\@gls@getbody#1#\def\@gls@body{#1}\@gls@gobbletonil}
```

Scoup up everything to \@nil and store in \@gls@rest:

```
4443 \def\@gls@gobbletonil#1\@nil{\def\@gls@rest{#1}}
```

`\xmakefirstuc` Expand argument once before applying `\makefirstuc` (added v1.01).

```
4444 \newcommand*\xmakefirstuc}[1]{%
```

```
4445 \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).

```
4446 \newcommand*\capitalisewords}[1]{%
```

```
4447 \def\gls@add@space{ }%
```

```
4448 \mfu@capitalisewords#1 \@nil\mfu@endcap
```

```
4449 %\gls@add@space\makefirstuc{##1}\def\gls@add@space{ }%
```

```
4450 }
```

```
4451 \def\mfu@capitalisewords#1 #2\mfu@endcap{%
```

```
4452 \def\mfu@cap@first{#1}%
```

```
4453 \def\mfu@cap@second{#2}%
```

```
4454 \gls@add@space
```

```
4455 \makefirstuc{#1}%
```

```
4456 \def\gls@add@space{ }%
```

```
4457 \ifx\mfu@cap@second\@nnil
```

```
4458 \let\next@mfu@cap\mfu@noop
```

```
4459 \else
```

```
4460 \let\next@mfu@cap\mfu@capitalisewords
```

```
4461 \fi
```

```
4462 \next@mfu@cap#2\mfu@endcap
```

```
4463 }
```

```
4464 \def\mfu@noop#1\mfu@endcap{ }
```

`\xcapitalisewords` Short-cut command:

```
4465 \newcommand*\xcapitalisewords}[1]{%
```

```
4466 \expandafter\capitalisewords\expandafter{#1}%
```

```
4467 }
```

3 Glossary Styles

3.1 Glossary hyper-navigation definitions (glossary-hypernav package)

Package Definition:

```
4468 \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 hypertarget in the event of multiple glossaries.

`\glsnavhyperlink[⟨type⟩]{⟨label⟩}{⟨text⟩}`

This command makes `⟨text⟩` a hyperlink to the glossary group whose label is given by `⟨label⟩` for the glossary given by `⟨type⟩`.

`\glsnavhyperlink`

```
4469 \newcommand*{\glsnavhyperlink}[3][\@glo@type]{%
4470   \edef\gls@grplabel{#2}\protected@edef\gls@grptitle{#3}%
4471   \@glslink{glsn:#1@#2}{#3}}
```

`\glsnavhypertarget[⟨type⟩]{⟨label⟩}{⟨text⟩}`

This command makes `⟨text⟩` a hypertarget for the glossary group whose label is given by `⟨label⟩` in the glossary given by `⟨type⟩`. If `⟨type⟩` is omitted, `\@glo@type` is used which is set by `\printglossary` to the current glossary label.

`\glsnavhypertarget`

```
4472 \newcommand*{\glsnavhypertarget}[3][\@glo@type]{%
  Add this group to the aux file for re-run check.
4473   \protected@write\@auxout{}{\string\gls@hypergroup{#1}{#2}}%
  Add the target.
4474   \@glstarget{glsn:#1@#2}{#3}%
  Check list of know groups to determine if a re-run is required.
4475   \expandafter\let
4476     \expandafter\@gls@list\csname @gls@hypergroup@list@#1\endcsname
  Iterate through list and terminate loop if this group is found.
4477   \@for\@gls@elem:=\@gls@list\do{%
4478     \ifthenelse{\equal{\@gls@elem}{#2}}{\@endfortrue}{}}%
  Check if list terminated prematurely.
4479   \if@endfor
4480   \else
    This group was not included in the list, so issue a warning.
4481     \GlossariesWarningNoLine{Navigation panel
4482       for glossary type ‘#1’^^Jmissing group ‘#2’}%
4483     \gdef\gls@hypergroup@rerun{%
4484       \GlossariesWarningNoLine{Navigation panel
4485         has changed. Rerun LaTeX}}%
4486     \fi
4487 }
```

`\gls@hypergroup@rerun` Give a warning at the end if re-run required

```
4488 \let\gls@hypergroup@rerun\relax
4489 \AtEndDocument{\gls@hypergroup@rerun}
```

`\@gls@hypergroup` This adds to (or creates) the command `\@gls@hypergroup@#1` (*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.

```

4490 \newcommand*{\@gls@hypergroup}[2]{%
4491 \ifundefined{\@gls@hypergroup@#1}{%
4492   \expandafter\edef\csname @gls@hypergroup@#1\endcsname{#2}%
4493 }{%
4494   \expandafter\let\expandafter\@gls@tmp
4495     \csname @gls@hypergroup@#1\endcsname
4496   \expandafter\edef\csname @gls@hypergroup@#1\endcsname{%
4497     \@gls@tmp,#2}%
4498 }%
4499 }

```

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`

```

4500 \newcommand*{\glsnavigation}{%
4501 \def\@gls@between{}%
4502 \ifundefined{\@gls@hypergroup@{\@glo@type}}{%
4503   \def\@gls@list{}%
4504 }{%
4505   \expandafter\let\expandafter\@gls@list
4506     \csname @gls@hypergroup@{\@glo@type}\endcsname
4507 }%
4508 \@for\@gls@tmp:=\@gls@list\do{%
4509   \@gls@between
4510   \glsnavhyperlink{\@gls@tmp}{\glsgetgrouptitle{\@gls@tmp}}%
4511   \let\@gls@between\glshypernavsep%
4512 }%
4513 }

```

`\glshypernavsep` Separator for the hyper navigation bar.

```

4514 \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`

```

4515 \newcommand*{\glssymbolnav}{%
4516 \glsnavhyperlink{glssymbols}{\glsgetgrouptitle{glssymbols}}%

```



```

4517 \glshypernavsep
4518 \glsnavhyperlink{glsnumbers}{\glsgetgrouptitle{glsnumbers}}%
4519 \glshypernavsep
4520 }

```

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.

```

4521 \ProvidesPackage{glossary-inline}[2012/09/21 v3.03 (NLCT)]

```

`inline` Define the inline style.

```

4522 \newglossarystyle{inline}{%
    Start of glossary sets up first empty separator between entries. (This is then
    changed by \glossaryentryfield)
4523   \renewenvironment{theglossary}{%
4524     {%
4525       \def\gls@inlinesep{}%
4526       \def\gls@inlinesubsep{}%
4527       \def\gls@inlinepostchild{}%
4528     }%
4529     {\glspostinline}%

```

No header:

```

4530 \renewcommand*{\glossaryheader}{}%
    No group headings (if heading is required, add \glsinlinedopostchild to
    start definition in case heading follows a child entry):
4531 \renewcommand*{\glsgroupheading}[1]{%

```

Just display separator followed by name and description:

```

4532 \renewcommand{\glossaryentryfield}[5]{%
4533   \glsinlinedopostchild
4534   \gls@inlinesep
4535   \def\glo@desc{##3}%
4536   \def\@no@post@desc{\nopostdesc}%
4537   \glsentryitem{##1}\glsinlinenameformat{##1}{##2}%
4538   \ifx\glo@desc\@no@post@desc
4539     \glsinlineemptydescformat{##4}{##5}%
4540   \else
4541     \ifstrempy{##3}%
4542       {\glsinlineemptydescformat{##4}{##5}}%
4543       {\glsinlinedescformat{##3}{##4}{##5}}%
4544   \fi
4545   \ifglshaschildren{##1}%
4546   {%
4547     \glsresetsubentrycounter
4548     \glsinlineparentchildseparator
4549     \def\gls@inlinesubsep{}%

```

```

4550     \def\gls@inlinepostchild{\glsinlinepostchild}%
4551   }%
4552   {}%
4553   \def\gls@inlinesep{\glsinlineseparator}%
4554   }%

```

Sub-entries display description:

```

4555   \renewcommand{\glossarysubentryfield}[6]{%
4556     \gls@inlinesubsep%
4557     \glsinlinesubnameformat{##2}{##3}%
4558     \glssubentryitem{##2}\glsinlinesubdescformat{##4}{##5}{##6}%
4559     \def\gls@inlinesubsep{\glsinlinesubseparator}%
4560   }%

```

Nothing special between groups:

```

4561   \renewcommand*{\glsgroupskip}{}%
4562 }

```

`\glsinlinedopostchild`

```

4563 \newcommand*{\glsinlinedopostchild}{%
4564   \gls@inlinepostchild
4565   \def\gls@inlinepostchild{}%
4566 }

```

`\glsinlineseparator` Separator to use between entries.

```

4567 \newcommand*{\glsinlineseparator}{;\space}

```

`\glsinlinesubseparator` Separator to use between sub-entries.

```

4568 \newcommand*{\glsinlinesubseparator}{,\space}

```

`\glsinlineparentchildseparator` Separator to use between parent and children.

```

4569 \newcommand*{\glsinlineparentchildseparator}{:\space}

```

`\glsinlinepostchild` Hook to use between child and next entry

```

4570 \newcommand*{\glsinlinepostchild}{}

```

`\glspostinline` Terminator for inline glossary.

```

4571 \newcommand*{\glspostinline}{\glspostdescription\space}

```

`\glsinlinenameformat` Formats the name of the entry (first argument label, second argument name):

```

4572 \newcommand*{\glsinlinenameformat}[2]{\glstarget{#1}{#2}}

```

`\glsinlinedescformat` Formats the entry's description, symbol and location list:

```

4573 \newcommand*{\glsinlinedescformat}[3]{\space#1}

```

`\glsinlineemptydescformat` Formats the entry's symbol and location list when the description is empty:

```

4574 \newcommand*{\glsinlineemptydescformat}[2]{}

```

`inlinesubnameformat` Formats the name of the subentry (first argument label, second argument name):

```
4575 \newcommand*{\glsinlinesubnameformat}[2]{\glstarget{#1}{}}
```

`inlinesubdescformat` Formats the subentry's description, symbol and location list:

```
4576 \newcommand*{\glsinlinesubdescformat}[3]{#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.

```
4577 \ProvidesPackage{glossary-list}[2012/09/21 v3.03 (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.

```
4578 \newglossarystyle{list}{%
```

Use description environment:

```
4579 \renewenvironment{theglossary}{%
```

```
4580 {\begin{description}}{\end{description}}%
```

No header at the start of the environment:

```
4581 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
4582 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries start a new item in the list:

```
4583 \renewcommand*{\glossaryentryfield}[5]{%
```

```
4584 \item[\glsentryitem{##1}\glstarget{##1}{##2}]
```

```
4585 ##3\glspostdescription\space ##5}%
```

Sub-entries continue on the same line:

```
4586 \renewcommand*{\glossarysubentryfield}[6]{%
```

```
4587 \glssubentryitem{##2}%
```

```
4588 \glstarget{##2}{\strut}##4\glspostdescription\space ##6.}%
```

```
4589 % \end{macrocode}
```

```
4590 % Add vertical space between groups:
```

```
4591 %\changes{3.03}{2012/09/21}{added check for glsnogroupskip}
```

```
4592 % \begin{macrocode}
```

```
4593 \renewcommand*{\glsgroupskip}{\ifglsgroupskip\else\indexspace\fi}%
```

```
4594 }
```

`listgroup` The listgroup style is like the list style, but the glossary groups have headings.

```
4595 \newglossarystyle{listgroup}{%
```

Base it on the list style:

```
4596 \glossarystyle{list}%
```

Each group has a heading:

```
4597 \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.

```
4598 \newglossarystyle{listhypergroup}{%
```

Base it on the list style:

```
4599 \glossarystyle{list}%
```

Add navigation links at the start of the environment:

```
4600 \renewcommand*{\glossaryheader}{%
```

```
4601 \item[\glsnavigation]}%
```

Each group has a heading with a hypertarget:

```
4602 \renewcommand*{\glsgroupheading}[1]{%
```

```
4603 \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.

```
4604 \newglossarystyle{altlist}{%
```

Base it on the list style:

```
4605 \glossarystyle{list}%
```

Main (level 0) entries start a new item in the list with a line break after the entry name:

```
4606 \renewcommand*{\glossaryentryfield}[5]{%
```

```
4607 \item[\glsentryitem{##1}\glstarget{##1}{##2}]\mbox{}\newline
```

```
4608 ##3\glspostdescription\space ##5}%
```

Sub-entries start a new paragraph:

```
4609 \renewcommand{\glossarysubentryfield}[6]{%
```

```
4610 \par
```

```
4611 \glssubentryitem{##2}%
```

```
4612 \glstarget{##2}{\strut}##4\glspostdescription\space ##6}%
```

```
4613 }
```

altlistgroup The altlistgroup glossary style is like the altlist style, but the glossary groups have headings.

```
4614 \newglossarystyle{altlistgroup}{%
```

Base it on the altlist style:

```
4615 \glossarystyle{altlist}%
```

Each group has a heading:

```
4616 \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.

```
4617 \newglossarystyle{altlisthypergroup}{%
```

Base it on the `altlist` style:

```
4618 \glossarystyle{altlist}%
```

Add navigation links at the start of the environment:

```
4619 \renewcommand*{\glossaryheader}{%
```

```
4620 \item[\glsnavigation]}%
```

Each group has a heading with a `hypertarget`:

```
4621 \renewcommand*{\glsgroupheading}[1]{%
```

```
4622 \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.

```
4623 \newglossarystyle{listdotted}{%
```

Base it on the `list` style:

```
4624 \glossarystyle{list}%
```

Each main (level 0) entry starts a new item:

```
4625 \renewcommand*{\glossaryentryfield}[5]{%
```

```
4626 \item[\makebox[\glslistdottedwidth][l]{%
```

```
4627 \glstryitem{##1}{\glstarget{##1}{##2}}%
```

```
4628 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##3}%
```

Sub entries have the same format as main entries:

```
4629 \renewcommand*{\glossarysubentryfield}[6]{%
```

```
4630 \item[\makebox[\glslistdottedwidth][l]{%
```

```
4631 \glssubentryitem{##2}}%
```

```
4632 \glstarget{##2}{##3}}%
```

```
4633 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##4}%
```

```
4634 }
```

`\glslistdottedwidth`

```
4635 \newlength\glslistdottedwidth
```

```
4636 \setlength{\glslistdottedwidth}{.5\hsize}
```

`sublistdotted` This style is similar to the `glostylelistdotted` style, except that the main entries just have the name displayed.

```
4637 \newglossarystyle{sublistdotted}{%
```

Base it on the `listdotted` style:

```
4638 \glossarystyle{listdotted}%
```

Main (level 0) entries just display the name:

```
4639 \renewcommand*{\glossaryentryfield}[5]{%
4640   \item[\glsentryitem{##1}\glstarget{##1}{##2}]}%
4641 }
```

3.4 Glossary Styles using longtable (the glossary-long package)

The glossary styles defined in the package used the longtable environment in the glossary.

```
4642 \ProvidesPackage{glossary-long}[2012/09/21 v3.03 (NLCT)]
```

Requires the package:

```
4643 \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`.)

```
4644 \@ifundefined{glsdescwidth}{%
4645   \newlength{glsdescwidth}
4646   \setlength{glsdescwidth}{0.6\hsize}
4647 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column.

```
4648 \@ifundefined{glspagelistwidth}{%
4649   \newlength{glspagelistwidth}
4650   \setlength{glspagelistwidth}{0.1\hsize}
4651 }{}
```

`long` The long glossary style command which uses the longtable environment:

```
4652 \newglossarystyle{long}{%
```

Use longtable with two columns:

```
4653   \renewenvironment{theglossary}%
4654     {\begin{longtable}\lp{glsdescwidth}}%
4655     {\end{longtable}}%
```

Do nothing at the start of the environment:

```
4656   \renewcommand*{\glossaryheader}{}%
```

No heading between groups:

```
4657   \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries displayed in a row:

```
4658   \renewcommand*{\glossaryentryfield}[5]{%
4659     \glsentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\}%
```

Sub entries displayed on the following row without the name:

```
4660   \renewcommand*{\glossarysubentryfield}[6]{%
4661     &
```

```

4662     \glssubentryitem{##2}%
4663     \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\}%

```

Blank row between groups:

```

4664     \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & \\fi}%
4665 }

```

longborder The longborder style is like the above, but with horizontal and vertical lines:

```

4666 \newglossarystyle{longborder}{%
    Base it on the glostylelong style:
4667     \glossarystyle{long}%
    Use longtable with two columns with vertical lines between each column:
4668     \renewenvironment{theglossary}{%
4669         \begin{longtable}{|l|p{\glstdescwidth}|}{\end{longtable}}%
    Place horizontal lines at the head and foot of the table:
4670     \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
4671 }

```

longheader The longheader style is like the long style but with a header:

```

4672 \newglossarystyle{longheader}{%
    Base it on the glostylelong style:
4673     \glossarystyle{long}%
    Set the table's header:
4674     \renewcommand*{\glossaryheader}{%
4675         \bfseries \entryname & \bfseries \descriptionname\\endhead}%
4676 }

```

longheaderborder The longheaderborder style is like the long style but with a header and border:

```

4677 \newglossarystyle{longheaderborder}{%
    Base it on the glostylelongborder style:
4678     \glossarystyle{longborder}%
    Set the table's header and add horizontal line to table's foot:
4679     \renewcommand*{\glossaryheader}{%
4680         \hline\bfseries \entryname & \bfseries \descriptionname\\hline
4681         \endhead
4682         \hline\endfoot}%
4683 }

```

long3col The long3col style is like long but with 3 columns

```

4684 \newglossarystyle{long3col}{%
    Use a longtable with 3 columns:
4685     \renewenvironment{theglossary}{%
4686         {\begin{longtable}{lp{\glstdescwidth}p{\glspagelistwidth}}}%
4687         {\end{longtable}}%

```

No table header:

```
4688 \renewcommand*{\glossaryheader}{}%
```

No headings between groups:

```
4689 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
4690 \renewcommand*{\glossaryentryfield}[5]{%
```

```
4691 \glstryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\\}%
```

Sub-entries on a separate row (no name, description in second column, page list in third column):

```
4692 \renewcommand*{\glossarysubentryfield}[6]{%
```

```
4693 &
```

```
4694 \glssubentryitem{##2}%
```

```
4695 \glstarget{##2}{\strut}##4 & ##6\\}%
```

Blank row between groups:

```
4696 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & &\\fi}%
```

```
4697 }
```

long3colborder The long3colborder style is like the long3col style but with a border:

```
4698 \newglossarystyle{long3colborder}{}%
```

Base it on the glostylelong3col style:

```
4699 \glossarystyle{long3col}%
```

Use a longtable with 3 columns with vertical lines around them:

```
4700 \renewenvironment{theglossary}%
```

```
4701 {\begin{longtable}{|l|p{\glsdescwidth}|p{\glspagelistwidth|}}%
```

```
4702 {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
4703 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
```

```
4704 }
```

long3colheader The long3colheader style is like long3col but with a header row:

```
4705 \newglossarystyle{long3colheader}{}%
```

Base it on the glostylelong3col style:

```
4706 \glossarystyle{long3col}%
```

Set the table's header:

```
4707 \renewcommand*{\glossaryheader}{}%
```

```
4708 \bfseries\entryname&\bfseries\descriptionname&
```

```
4709 \bfseries\pagelistname\\endhead}%
```

```
4710 }
```

ong3colheaderborder The long3colheaderborder style is like the above but with a border

```
4711 \newglossarystyle{long3colheaderborder}{}%
```


Base it on the `glostylelong3colborder` style:

```
4712 \glossarystyle{long3colborder}{%
```

Set the table's header and add horizontal line at table's foot:

```
4713 \renewcommand*{\glossaryheader}{%  
4714 \hline  
4715 \bfseries\entryname&\bfseries\descriptionname&  
4716 \bfseries\pagelistname\\ \hline\endhead  
4717 \hline\endfoot}%  
4718 }
```

`long4col` The `long4col` style has four columns where the third column contains the value of the associated symbol key.

```
4719 \newglossarystyle{long4col}{%
```

Use a `longtable` with 4 columns:

```
4720 \renewenvironment{theglossary}%  
4721 {\begin{longtable}{llll}}%  
4722 {\end{longtable}}%
```

No table header:

```
4723 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
4724 \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):

```
4725 \renewcommand*{\glossaryentryfield}[5]{%  
4726 \glsentryitem{##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):

```
4727 \renewcommand*{\glossarysubentryfield}[6]{%  
4728 &  
4729 \glssubentryitem{##2}%  
4730 \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%
```

Blank row between groups:

```
4731 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & & \\\fi}%  
4732 }
```

`long4colheader` The `long4colheader` style is like `long4col` but with a header row.

```
4733 \newglossarystyle{long4colheader}{%
```

Base it on the `glostylelong4col` style:

```
4734 \glossarystyle{long4col}%
```

Table has a header:

```
4735 \renewcommand*{\glossaryheader}{%  
4736 \bfseries\entryname&\bfseries\descriptionname&  
4737 \bfseries \symbolname&
```

```

4738 \bfseries\pagelistname\\\endhead}%
4739 }

```

long4colborder The long4colborder style is like long4col but with a border.

```

4740 \newglossarystyle{long4colborder}{%

```

Base it on the glostylelong4col style:

```

4741 \glossarystyle{long4col}%

```

Use a longtable with 4 columns surrounded by vertical lines:

```

4742 \renewenvironment{theglossary}%
4743 {\begin{longtable}{|l|l|l|l|}}%
4744 {\end{longtable}}%

```

Add horizontal lines to the head and foot of the table:

```

4745 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
4746 }

```

long4colheaderborder The long4colheaderborder style is like the above but with a border.

```

4747 \newglossarystyle{long4colheaderborder}{%

```

Base it on the glostylelong4col style:

```

4748 \glossarystyle{long4col}%

```

Use a longtable with 4 columns surrounded by vertical lines:

```

4749 \renewenvironment{theglossary}%
4750 {\begin{longtable}{|l|l|l|l|}}%
4751 {\end{longtable}}%

```

Add table header and horizontal line at the table's foot:

```

4752 \renewcommand*{\glossaryheader}{%
4753 \hline\bfseries\entryname&\bfseries\descriptionname&
4754 \bfseries \symbolname&
4755 \bfseries\pagelistname\\\hline\endhead\hline\endfoot}%
4756 }

```

altlong4col The altlong4col style is like the long4col style but can have multiline descriptions and page lists.

```

4757 \newglossarystyle{altlong4col}{%

```

Base it on the glostylelong4col style:

```

4758 \glossarystyle{long4col}%

```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```

4759 \renewenvironment{theglossary}%
4760 {\begin{longtable}{lp{\glsdescwidth}lp{\glspagelistwidth}}}%
4761 {\end{longtable}}%
4762 }

```

altlong4colheader The altlong4colheader style is like altlong4col but with a header row.

```

4763 \newglossarystyle{altlong4colheader}{%

```

Base it on the `glostylelong4colheader` style:

```
4764 \glossarystyle{long4colheader}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
4765 \renewenvironment{theglossary}%  
4766   {\begin{longtable}{lp{\glsgdescwidth}lp{\glspagelistwidth}}}%  
4767   {\end{longtable}}%  
4768 }
```

`altlong4colborder` The `altlong4colborder` style is like `altlong4col` but with a border.

```
4769 \newglossarystyle{altlong4colborder}{%
```

Base it on the `glostylelong4colborder` style:

```
4770 \glossarystyle{long4colborder}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
4771 \renewenvironment{theglossary}%  
4772   {\begin{longtable}{|lp{\glsgdescwidth}|lp{\glspagelistwidth}|}%  
4773   {\end{longtable}}%  
4774 }
```

`altlong4colheaderborder` The `altlong4colheaderborder` style is like the above but with a header as well as a border.

```
4775 \newglossarystyle{altlong4colheaderborder}{%
```

Base it on the `glostylelong4colheaderborder` style:

```
4776 \glossarystyle{long4colheaderborder}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
4777 \renewenvironment{theglossary}%  
4778   {\begin{longtable}{|lp{\glsgdescwidth}|lp{\glspagelistwidth}|}%  
4779   {\end{longtable}}%  
4780 }
```

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.

```
4781 \ProvidesPackage{glossary-longragged}[2012/09/21 v3.03 (NLCT)]
```

Requires the package:

```
4782 \RequirePackage{array}
```

Requires the package:

```
4783 \RequirePackage{longtable}
```

`\glsdescwidth` This is a length that governs the width of the description column. This may have already been defined.

```
4784 \@ifundefined{glsdescwidth}{%
4785   \newlength\glsdescwidth
4786   \setlength{\glsdescwidth}{0.6\hsize}
4787 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

```
4788 \@ifundefined{glspagelistwidth}{%
4789   \newlength\glspagelistwidth
4790   \setlength{\glspagelistwidth}{0.1\hsize}
4791 }{}
```

`longragged` The longragged glossary style is like the long but uses ragged right formatting for the description column.

```
4792 \newglossarystyle{longragged}{%
```

Use longtable with two columns:

```
4793   \renewenvironment{theglossary}%
4794     {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}}}%
4795     {\end{longtable}}%
```

Do nothing at the start of the environment:

```
4796   \renewcommand*{\glossaryheader}{}%
```

No heading between groups:

```
4797   \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries displayed in a row:

```
4798   \renewcommand*{\glossaryentryfield}[5]{%
4799     \glstentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%
4800     \tabularnewline}%
```

Sub entries displayed on the following row without the name:

```
4801   \renewcommand*{\glossarysubentryfield}[6]{%
4802     &
4803     \glssubentryitem{##2}%
4804     \glstarget{##2}{\strut}##4\glspostdescription\space ##6%
4805     \tabularnewline}%
```

Blank row between groups:

```
4806   \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & \tabularnewline\fi}%
4807 }
```

`longraggedborder` The longraggedborder style is like the above, but with horizontal and vertical lines:

```
4808 \newglossarystyle{longraggedborder}{%
```

Base it on the `glostylelongragged` style:

```
4809   \glossarystyle{longragged}%
```

Use longtable with two columns with vertical lines between each column:

```
4810 \renewenvironment{theglossary}{%  
4811 \begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|}%  
4812 {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
4813 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%  
4814 }
```

`longraggedheader` The longraggedheader style is like the longragged style but with a header:

```
4815 \newglossarystyle{longraggedheader}{%
```

Base it on the `glostylelongragged` style:

```
4816 \glossarystyle{longragged}%
```

Set the table's header:

```
4817 \renewcommand*{\glossaryheader}{%  
4818 \bfseries \entryname & \bfseries \descriptionname  
4819 \tabularnewline\endhead}%  
4820 }
```

`longraggedheaderborder` The longraggedheaderborder style is like the longragged style but with a header and border:

```
4821 \newglossarystyle{longraggedheaderborder}{%
```

Base it on the `glostylelongraggedborder` style:

```
4822 \glossarystyle{longraggedborder}%
```

Set the table's header and add horizontal line to table's foot:

```
4823 \renewcommand*{\glossaryheader}{%  
4824 \hline\bfseries \entryname & \bfseries \descriptionname  
4825 \tabularnewline\hline  
4826 \endhead  
4827 \hline\endfoot}%  
4828 }
```

`longragged3col` The longragged3col style is like longragged but with 3 columns

```
4829 \newglossarystyle{longragged3col}{%
```

Use a longtable with 3 columns:

```
4830 \renewenvironment{theglossary}{%  
4831 {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}%  
4832 >{\raggedright}p{\gls pagelistwidth}}}%  
4833 {\end{longtable}}%
```

No table header:

```
4834 \renewcommand*{\glossaryheader}{}%
```

No headings between groups:

```
4835 \renewcommand*{\gls groupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
4836 \renewcommand*{\glossaryentryfield}[5]{%
4837   \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):

```
4838 \renewcommand*{\glossarysubentryfield}[6]{%
4839   &
4840   \glssubentryitem{##2}%
4841   \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%
```

Blank row between groups:

```
4842 \renewcommand*{\glsnoglobskip}{\ifglsnogroupskip\else & \&\tabularnewline\fi}%
4843 }
```

`longragged3colborder` The `longragged3colborder` style is like the `longragged3col` style but with a border:

```
4844 \newglossarystyle{longragged3colborder}{%
```

Base it on the `glostylelongragged3col` style:

```
4845 \glossarystyle{longragged3col}%
```

Use a `longtable` with 3 columns with vertical lines around them:

```
4846 \renewenvironment{theglossary}%
4847   {\begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|%
4848    >{\raggedright}p{\glspagelistwidth}|}%
4849   {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
4850 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
4851 }
```

`longragged3colheader` The `longragged3colheader` style is like `longragged3col` but with a header row:

```
4852 \newglossarystyle{longragged3colheader}{%
```

Base it on the `glostylelongragged3col` style:

```
4853 \glossarystyle{longragged3col}%
```

Set the table's header:

```
4854 \renewcommand*{\glossaryheader}{%
4855   \bfseries\entryname&\bfseries\descriptionname&
4856   \bfseries\pagelistname\tabularnewline\endhead}%
4857 }
```

`longragged3colheaderborder` The `longragged3colheaderborder` style is like the above but with a border

```
4858 \newglossarystyle{longragged3colheaderborder}{%
```

Base it on the `glostylelongragged3colborder` style:

```
4859 \glossarystyle{longragged3colborder}%
```

Set the table's header and add horizontal line at table's foot:

```
4860 \renewcommand*{\glossaryheader}{%
4861 \hline
4862 \bfseries\entryname&\bfseries\descriptionname&
4863 \bfseries\pagelistname\tabularnewline\hline\endhead
4864 \hline\endfoot}%
4865 }
```

`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.

```
4866 \newglossarystyle{altlongragged4col}{%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
4867 \renewenvironment{theglossary}%
4868 {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%
4869 >{\raggedright}p{\glspagelistwidth}}}%
4870 {\end{longtable}}%
```

No table header:

```
4871 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
4872 \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):

```
4873 \renewcommand*{\glossaryentryfield}[5]{%
4874 \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):

```
4875 \renewcommand*{\glossarysubentryfield}[6]{%
4876 &
4877 \glssubentryitem{##2}%
4878 \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%
```

Blank row between groups:

```
4879 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & & \tabularnewline\fi}%
4880 }
```

`ongragged4colheader` The `altlongragged4colheader` style is like `altlongragged4col` but with a header row.

```
4881 \newglossarystyle{altlongragged4colheader}{%
```

Base it on the `glostylealtlongragged4col` style:

```
4882 \glossarystyle{altlongragged4col}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
4883 \renewenvironment{theglossary}%
4884   {\begin{longtable}{l>{\raggedright}p{\glsgdescwidth}l%
4885    >{\raggedright}p{\glspagelistwidth}}}%
4886   {\end{longtable}}%
```

Table has a header:

```
4887 \renewcommand*{\glossaryheader}{%
4888   \bfseries\entryname&\bfseries\descriptionname&
4889   \bfseries \symbolname&
4890   \bfseries\pagelistname\tabularnewline\endhead}%
4891 }
```

`ongragged4colborder` The `altlongragged4colborder` style is like `altlongragged4col` but with a border.

```
4892 \newglossarystyle{altlongragged4colborder}{%
```

Base it on the `glostylealtlongragged4col` style:

```
4893   \glossarystyle{altlongragged4col}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
4894 \renewenvironment{theglossary}%
4895   {\begin{longtable}{|l|>{\raggedright}p{\glsgdescwidth}|l|}%
4896    >{\raggedright}p{\glspagelistwidth}|}%
4897   {\end{longtable}}%
```

Add horizontal lines to the head and foot of the table:

```
4898 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
4899 }
```

`ged4colheaderborder` The `altlongragged4colheaderborder` style is like the above but with a header as well as a border.

```
4900 \newglossarystyle{altlongragged4colheaderborder}{%
```

Base it on the `glostylealtlongragged4col` style:

```
4901   \glossarystyle{altlongragged4col}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
4902 \renewenvironment{theglossary}%
4903   {\begin{longtable}{|l|>{\raggedright}p{\glsgdescwidth}|l|}%
4904    >{\raggedright}p{\glspagelistwidth}|}%
4905   {\end{longtable}}%
```

Add table header and horizontal line at the table's foot:

```
4906 \renewcommand*{\glossaryheader}{%
4907   \hline\bfseries\entryname&\bfseries\descriptionname&
4908   \bfseries \symbolname&
4909   \bfseries\pagelistname\tabularnewline\hline\endhead
4910   \hline\endfoot}%
4911 }
```


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.

```
4912 \ProvidesPackage{glossary-mcols}[2012/05/21 v1.0 (NLCT)]
```

Required packages:

```
4913 \RequirePackage{multicol}
```

```
4914 \RequirePackage{glossary-tree}
```

`\glsmcols` Define macro in which to store the number of columns. (Defaults to 2.)

```
4915 \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.)

```
4916 \newglossarystyle{mcolindex}{%
4917   \glossarystyle{index}%
4918   \renewenvironment{theglossary}%
4919     {%
4920       \begin{multicols}{2}
4921       \setlength{\parindent}{0pt}%
4922       \setlength{\parskip}{0pt plus 0.3pt}%
4923       \let\item\@idxitem}%
4924     {\end{multicols}}%
4925 }
```

`mcolindexgroup` As `mcolindex` but has headings:

```
4926 \newglossarystyle{mcolindexgroup}{%
4927   \glossarystyle{mcolindex}%
4928   \renewcommand*\glsgroupheading[1]{%
4929     \item\textbf{\glsgroupheading{##1}}\indexspace}%
4930 }
```

`mcolindexhypergroup` The `mcolindexhypergroup` style is like the `mcolindexgroup` style but has hyper navigation.

```
4931 \newglossarystyle{mcolindexhypergroup}{%
```

Base it on the `glostylemcolindex` style:

```
4932   \glossarystyle{mcolindex}%
```

Put navigation links to the groups at the start of the glossary:

```
4933   \renewcommand*\glossaryheader{%
4934     \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.

```
4935   \renewcommand*\glsgroupheading[1]{%
4936     \item\textbf{\glsnavigationhypertarget{##1}}{\glsgroupheading{##1}}}%
4937   \indexspace}%
4938 }
```

mcoltree Multi-column index style. Same as the tree, but puts the glossary in multiple columns.

```
4939 \newglossarystyle{mcoltree}{%
4940   \glossarystyle{tree}%
4941   \renewenvironment{theglossary}%
4942   {%
4943     \begin{multicols}{2}
4944     \setlength{\parindent}{0pt}%
4945     \setlength{\parskip}{0pt plus 0.3pt}%
4946   }%
4947   {\end{multicols}}%
4948 }
```

mcoltreegroup Like the mcoltree style but the glossary groups have headings.

```
4949 \newglossarystyle{mcoltreegroup}{%
  Base it on the glostylemcoltree style:
4950   \glossarystyle{mcoltree}%
  Each group has a heading (in bold) followed by a vertical gap):
4951   \renewcommand{\glsgroupheading}[1]{\par
4952     \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
4953 }
```

mcoltreehypergroup The mcoltreehypergroup style is like the treegroup style, but has a set of links to the groups at the start of the glossary.

```
4954 \newglossarystyle{mcoltreehypergroup}{%
  Base it on the glostylemcoltree style:
4955   \glossarystyle{mcoltree}%
  Put navigation links to the groups at the start of the theglossary environment:
4956   \renewcommand*{\glsgroupheading}[1]{%
4957     \par\noindent\textbf{\glsnavigation}\par\indexspace}%
  Each group has a heading (in bold with a target) followed by a vertical gap):
4958   \renewcommand*{\glsgroupheading}[1]{%
4959     \par\noindent
4960     \textbf{\glsnavigationhypertarget{##1}{\glsgrouptitle{##1}}}\par
4961     \indexspace}%
4962 }
```

mcoltreename Multi-column index style. Same as the treename, but puts the glossary in multiple columns.

```
4963 \newglossarystyle{mcoltreename}{%
4964   \glossarystyle{treename}%
4965   \renewenvironment{theglossary}%
4966   {%
4967     \begin{multicols}{2}
4968     \setlength{\parindent}{0pt}%
4969   }
```

```

4969     \setlength{\parskip}{0pt plus 0.3pt}%
4970   }%
4971   {\end{multicols}}}%
4972 }

```

mcoltreenonamegroup Like the mcoltreenoname style but the glossary groups have headings.

```

4973 \newglossarystyle{mcoltreenonamegroup}{%
    Base it on the glostylemcoltreenoname style:
4974   \glossarystyle{mcoltreenoname}%
    Give each group a heading:
4975   \renewcommand{\glsgroupheading}[1]{\par
4976     \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
4977 }

```

treenonamehypergroup The mcoltreenonamehypergroup style is like the mcoltreenonamegroup style, but has a set of links to the groups at the start of the glossary.

```

4978 \newglossarystyle{mcoltreenonamehypergroup}{%
    Base it on the glostylemcoltreenoname style:
4979   \glossarystyle{mcoltreenoname}%
    Put navigation links to the groups at the start of the theglossary environment:
4980   \renewcommand*{\glossaryheader}{%
4981     \par\noindent\textbf{\glsnavigation}\par\indexspace}%
    Each group has a heading (in bold with a target) followed by a vertical gap):
4982   \renewcommand*{\glsgroupheading}[1]{%
4983     \par\noindent
4984     \textbf{\glsnavigationhypertarget{##1}{\glsgrouptitle{##1}}}\par
4985     \indexspace}%
4986 }

```

mcolalmtree Multi-column index style. Same as the almtree, but puts the glossary in multiple columns.

```

4987 \newglossarystyle{mcolalmtree}{%
4988   \glossarystyle{almtree}%
4989   \renewenvironment{theglossary}%
4990   {%
4991     \begin{multicols}{2}%
4992     \def\@gls@prevlevel{-1}%
4993     \mbox{}\par
4994   }%
4995   {\par\end{multicols}}%
4996 }

```

mcolalmtreegroup Like the mcolalmtree style but the glossary groups have headings.

```

4997 \newglossarystyle{mcolalmtreegroup}{%

```

Base it on the `glostylemcolalttree` style:

```
4998 \glossarystyle{mcolalttree}%
Give each group a heading.
4999 \renewcommand{\glsgroupheading}[1]{\par
5000 \def\@gls@prevlevel{-1}%
5001 \hangindent0pt\relax
5002 \parindent0pt\relax
5003 \textbf{\glsgrouptitle{##1}}\par\indexspace}%
5004 }
```

`colalttreehypergroup` The `mcolalttreehypergroup` style is like the `mcolalttreegroup` style, but has a set of links to the groups at the start of the glossary.

```
5005 \newglossarystyle{mcolalttreehypergroup}{%
```

Base it on the `glostylemcolalttree` style:

```
5006 \glossarystyle{mcolalttree}%
Put the navigation links in the header
5007 \renewcommand*{\glossaryheader}{%
5008 \par
5009 \def\@gls@prevlevel{-1}%
5010 \hangindent0pt\relax
5011 \parindent0pt\relax
5012 \textbf{\glsnavigation}\par\indexspace}%
Put a hypertarget at the start of each group
```

```
5013 \renewcommand*{\glsgroupheading}[1]{%
5014 \par
5015 \def\@gls@prevlevel{-1}%
5016 \hangindent0pt\relax
5017 \parindent0pt\relax
5018 \textbf{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}\par
5019 \indexspace}}
```

3.7 Glossary Styles using supertabular environment (glossary-super package)

The glossary styles defined in the package use the `supertabular` environment.

```
5020 \ProvidesPackage{glossary-super}[2012/09/21 v3.03 (NLCT)]
```

Requires the package:

```
5021 \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.

```
5022 \@ifundefined{glsdescwidth}{%
5023 \newlength\glsdescwidth
5024 \setlength{\glsdescwidth}{0.6\hsize}
5025 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined if has been loaded.

```
5026 \@ifundefined{glspagelistwidth}{%
5027   \newlength{glspagelistwidth}
5028   \setlength{glspagelistwidth}{0.1\hsize}
5029 }{}
```

super The super glossary style uses the supertabular environment (it uses lengths defined in the package.)

```
5030 \newglossarystyle{super}{%
```

Put the glossary in a supertabular environment with two columns and no head or tail:

```
5031   \renewenvironment{theglossary}%
5032     {\tablehead{}\tabletail}%
5033     \begin{supertabular}{lp{glstdescwidth}}%
5034     {\end{supertabular}}%
```

Do nothing at the start of the table:

```
5035   \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
5036   \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries put in a row (name in first column, description and page list in second column):

```
5037   \renewcommand*{\glossaryentryfield}[5]{%
5038     \glstryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\}%
```

Sub entries put in a row (no name, description and page list in second column):

```
5039   \renewcommand*{\glossarysubentryfield}[6]{%
5040     &
5041     \glssubentryitem{##2}%
5042     \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\}%
```

Blank row between groups:

```
5043   \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & \\fi}%
5044 }
```

superborder The superborder style is like the above, but with horizontal and vertical lines:

```
5045 \newglossarystyle{superborder}{%
```

Base it on the glostylesuper style:

```
5046   \glossarystyle{super}%
```

Put the glossary in a supertabular environment with two columns and a horizontal line in the head and tail:

```
5047   \renewenvironment{theglossary}%
5048     {\tablehead{\hline}\tabletail{\hline}%
5049     \begin{supertabular}{|lp{glstdescwidth}|}%
5050     {\end{supertabular}}%
5051 }
```

superheader The superheader style is like the super style, but with a header:

```
5052 \newglossarystyle{superheader}{%  
    Base it on the glostylesuper style:  
5053   \glossarystyle{super}%  
    Put the glossary in a supertabular environment with two columns, a header and  
    no tail:  
5054 \renewenvironment{theglossary}%  
5055   {\tablehead{\bfseries \entryname & \bfseries \descriptionname\\}%  
5056    \tabletail{}}%  
5057   \begin{supertabular}{lp{\glsdescwidth}}}%  
5058   {\end{supertabular}}}%  
5059 }
```

superheaderborder The superheaderborder style is like the super style but with a header and border:

```
5060 \newglossarystyle{superheaderborder}{%  
    Base it on the glostylesuper style:  
5061   \glossarystyle{super}%  
    Put the glossary in a supertabular environment with two columns, a header and  
    horizontal lines above and below the table:  
5062   \renewenvironment{theglossary}%  
5063     {\tablehead{\hline\bfseries \entryname &  
5064       \bfseries \descriptionname\\ \hline}%  
5065      \tabletail{\hline}  
5066      \begin{supertabular}{|l|p{\glsdescwidth}|}%  
5067      {\end{supertabular}}}%  
5068 }
```

super3col The super3col style is like the super style, but with 3 columns:

```
5069 \newglossarystyle{super3col}{%  
    Put the glossary in a supertabular environment with three columns and no head  
    or tail:  
5070   \renewenvironment{theglossary}%  
5071     {\tablehead{} \tabletail{}}%  
5072     \begin{supertabular}{lp{\glsdescwidth}p{\glspagelistwidth}}}%  
5073     {\end{supertabular}}}%  
    Do nothing at the start of the table:  
5074   \renewcommand*{\glossaryheader}{}%  
    No group headings:  
5075   \renewcommand*{\glsgroupheading}[1]{}%  
    Main (level 0) entries on a row (name in first column, description in second  
    column, page list in last column):  
5076   \renewcommand*{\glossaryentryfield}[5]{}%  
5077     \glsentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\\}%
```

Sub entries on a row (no name, description in second column, page list in last column):

```
5078 \renewcommand*{\glossarysubentryfield}[6]{%
5079     &
5080     \glssubentryitem{##2}%
5081     \glstarget{##2}{\strut}##4 & ##6\\}%
```

Blank row between groups:

```
5082 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & &\\fi}%
5083 }
```

super3colborder The super3colborder style is like the super3col style, but with a border:

```
5084 \newglossarystyle{super3colborder}{%
    Base it on the glostylesuper3col style:
5085 \glossarystyle{super3col}%
    Put the glossary in a supertabular environment with three columns and a horizontal line in the head and tail:
5086 \renewenvironment{theglossary}%
5087     {\tablehead{\hline}\tabletail{\hline}%
5088     \begin{supertabular}{|l|p{\glstdescwidth}|p{\glspagelistwidth}|}%
5089     {\end{supertabular}}%
5090 }
```

super3colheader The super3colheader style is like the super3col style but with a header row:

```
5091 \newglossarystyle{super3colheader}{%
    Base it on the glostylesuper3col style:
5092 \glossarystyle{super3col}%
    Put the glossary in a supertabular environment with three columns, a header and no tail:
5093 \renewenvironment{theglossary}%
5094     {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
5095     \bfseries\pagelistname\\}\tabletail{}}%
5096     \begin{supertabular}{lp{\glstdescwidth}p{\glspagelistwidth}}}%
5097     {\end{supertabular}}%
5098 }
```

super3colheaderborder The super3colheaderborder style is like the super3col style but with a header and border:

```
5099 \newglossarystyle{super3colheaderborder}{%
    Base it on the glostylesuper3colborder style:
5100 \glossarystyle{super3colborder}%
    Put the glossary in a supertabular environment with three columns, a header with horizontal lines and a horizontal line in the tail:
5101 \renewenvironment{theglossary}%
5102     {\tablehead{\hline
```

```

5103      \bfseries\entryname&\bfseries\descriptionname&
5104      \bfseries\pagelistname\\\hline}%
5105      \tabletail{\hline}%
5106      \begin{supertabular}{|l|p{\glsgdescwidth}|p{\glspagelistwidth}|}%
5107      {\end{supertabular}}}%
5108 }

```

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.

```
5109 \newglossarystyle{super4col}{%
```

Put the glossary in a supertabular environment with four columns and no head or tail:

```

5110   \renewenvironment{theglossary}%
5111   {\tablehead{\tabletail}%
5112   \begin{supertabular}{|l|l|l|l|}%
5113   \end{supertabular}}%

```

Do nothing at the start of the table:

```
5114   \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
5115   \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:

```

5116   \renewcommand*{\glossaryentryfield}[5]{%
5117   \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:

```

5118   \renewcommand*{\glossarysubentryfield}[6]{%
5119   &
5120   \glssubentryitem{##2}%
5121   \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%

```

Blank row between groups:

```

5122   \renewcommand*{\glsgroupskip}{\ifglsgnigroupskip\else & & \\\fi}%
5123 }

```

super4colheader The **super4colheader** style is like the **super4col** but with a header row.

```
5124 \newglossarystyle{super4colheader}{%
```

Base it on the **glostylesuper4col** style:

```
5125   \glossarystyle{super4col}%
```

Put the glossary in a supertabular environment with four columns, a header and no tail:

```

5126   \renewenvironment{theglossary}%
5127   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
5128   \bfseries\symbolname &

```



```

5129      \bfseries\pagelistname\\}%
5130      \tabletail{}}%
5131      \begin{supertabular}{|l|l|l|l|}%
5132      {\end{supertabular}}}%
5133 }

```

super4colborder The super4colborder style is like the super4col but with a border.

```
5134 \newglossarystyle{super4colborder}{%
```

Base it on the glostylesuper4col style:

```
5135 \glossarystyle{super4col}%
```

Put the glossary in a supertabular environment with four columns and a horizontal line in the head and tail:

```

5136 \renewenvironment{theglossary}%
5137 {\tablehead{\hline}\tabletail{\hline}%
5138 \begin{supertabular}{|l|l|l|l|}%
5139 {\end{supertabular}}}%
5140 }

```

per4colheaderborder The super4colheaderborder style is like the super4col but with a header and border.

```
5141 \newglossarystyle{super4colheaderborder}{%
```

Base it on the glostylesuper4col style:

```
5142 \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:

```

5143 \renewenvironment{theglossary}%
5144 {\tablehead{\hline\bfseries\entryname&\bfseries\descriptionname&
5145 \bfseries\symbolname &
5146 \bfseries\pagelistname\\hline}\tabletail{\hline}%
5147 \begin{supertabular}{|l|l|l|l|}%
5148 {\end{supertabular}}}%
5149 }

```

altsuper4col The altsuper4col glossary style is like super4col but has provision for multiline descriptions.

```
5150 \newglossarystyle{altsuper4col}{%
```

Base it on the glostylesuper4col style:

```
5151 \glossarystyle{super4col}%
```

Put the glossary in a supertabular environment with four columns and no head or tail:

```

5152 \renewenvironment{theglossary}%
5153 {\tablehead{}\tabletail{}}%
5154 \begin{supertabular}{lp{\glstdescwidth}lp{\glspagelistwidth}}}%
5155 {\end{supertabular}}}%
5156 }

```

altsuper4colheader The altsuper4colheader style is like the altsuper4col but with a header row.

```

5157 \newglossarystyle{altsuper4colheader}{%
    Base it on the glostylesuper4colheader style:
5158   \glossarystyle{super4colheader}%

    Put the glossary in a supertabular environment with four columns, a header and
    no tail:
5159   \renewenvironment{theglossary}%
5160     {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
5161       \bfseries\symbolname &
5162       \bfseries\pagelistname\\}\tabletail{}}%
5163     \begin{supertabular}{lp{\glsgdescwidth}lp{\glspagelistwidth}}}%
5164     {\end{supertabular}}}%
5165 }
```

altsuper4colborder The altsuper4colborder style is like the altsuper4col but with a border.

```

5166 \newglossarystyle{altsuper4colborder}{%
    Base it on the glostylesuper4colborder style:
5167   \glossarystyle{super4colborder}%

    Put the glossary in a supertabular environment with four columns and a horizontal
    line in the head and tail:
5168   \renewenvironment{theglossary}%
5169     {\tablehead{\hline}\tabletail{\hline}%
5170     \begin{supertabular}%
5171       {\lllp{\glsgdescwidth}lllp{\glspagelistwidth}}}%
5172     {\end{supertabular}}}%
5173 }
```

per4colheaderborder The altsuper4colheaderborder style is like the altsuper4col but with a header and border.

```

5174 \newglossarystyle{altsuper4colheaderborder}{%
    Base it on the glostylesuper4colheaderborder style:
5175   \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:
5176   \renewenvironment{theglossary}%
5177     {\tablehead{\hline
5178       \bfseries\entryname &
5179       \bfseries\descriptionname &
5180       \bfseries\symbolname &
5181       \bfseries\pagelistname\\\hline}%
5182     \tabletail{\hline}%
5183     \begin{supertabular}%
5184       {\lllp{\glsgdescwidth}lllp{\glspagelistwidth}}}%
5185     {\end{supertabular}}}%
5186 }
```

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.

5187 \ProvidesPackage{glossary-superragged}[2012/09/21 v3.03 (NLCT)]

Requires the package:

5188 \RequirePackage{array}

Requires the package:

5189 \RequirePackage{supertabular}

`\glsdescwidth` This is a length that governs the width of the description column. This may already have been defined.

```
5190 \@ifundefined{glsdescwidth}{%
5191   \newlength{glsdescwidth
5192   \setlength{glsdescwidth}{0.6\hspace}
5193 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

```
5194 \@ifundefined{glspagelistwidth}{%
5195   \newlength{glspagelistwidth
5196   \setlength{glspagelistwidth}{0.1\hspace}
5197 }{}
```

`superragged` The superragged glossary style uses the supertabular environment.

5198 \newglossarystyle{superragged}{%

Put the glossary in a supertabular environment with two columns and no head or tail:

```
5199   \renewenvironment{theglossary}%
5200     {\tablehead{}\tabletail}%
5201     \begin{supertabular}{l>{\raggedright}p{glsdescwidth}}}%
5202     {\end{supertabular}}%
```

Do nothing at the start of the table:

5203 \renewcommand*{\glossaryheader}{}%

No group headings:

5204 \renewcommand*{\glsgroupheading}[1]{}%

Main (level 0) entries put in a row (name in first column, description and page list in second column):

```
5205 \renewcommand*{\glossaryentryfield}[5]{%
5206   \glstryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%
5207   \tabularnewline}%

```

Sub entries put in a row (no name, description and page list in second column):

```
5208 \renewcommand*{\glossarysubentryfield}[6]{%
5209     &
5210     \glssubentryitem{##2}%
5211     \glstarget{##2}{\strut}##4\glspostdescription\space ##6%
5212     \tabularnewline}%
```

Blank row between groups:

```
5213 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & \tabularnewline\fi}%
5214 }
```

superraggedborder The superraggedborder style is like the above, but with horizontal and vertical lines:

```
5215 \newglossarystyle{superraggedborder}{%
```

Base it on the glostylesuperragged style:

```
5216 \glossarystyle{superragged}%
```

Put the glossary in a supertabular environment with two columns and a horizontal line in the head and tail:

```
5217 \renewenvironment{theglossary}%
5218     {\tablehead{\hline}\tabletail{\hline}%
5219     \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}|}}%
5220     {\end{supertabular}}%
5221 }
```

superraggedheader The superraggedheader style is like the super style, but with a header:

```
5222 \newglossarystyle{superraggedheader}{%
```

Base it on the glostylesuperragged style:

```
5223 \glossarystyle{superragged}%
```

Put the glossary in a supertabular environment with two columns, a header and no tail:

```
5224 \renewenvironment{theglossary}%
5225     {\tablehead{\bfseries \entryname & \bfseries \descriptionname
5226         \tabularnewline}%
5227     \tabletail{}}%
5228     \begin{supertabular}{l>{\raggedright}p{\glsgdescwidth}}}%
5229     {\end{supertabular}}%
5230 }
```

rraggedheaderborder The superraggedheaderborder style is like the superragged style but with a header and border:

```
5231 \newglossarystyle{superraggedheaderborder}{%
```

Base it on the glostylesuper style:

```
5232 \glossarystyle{superragged}%
```

Put the glossary in a supertabular environment with two columns, a header and horizontal lines above and below the table:

```

5233 \renewenvironment{theglossary}%
5234   {\tablehead{\hline\bfseries \entryname &
5235     \bfseries \descriptionname\tabularnewline\hline}%
5236   \tabletail{\hline}
5237   \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}|}}%
5238   {\end{supertabular}}%
5239 }

```

superragged3col The superragged3col style is like the superragged style, but with 3 columns:

```
5240 \newglossarystyle{superragged3col}{%
```

Put the glossary in a supertabular environment with three columns and no head or tail:

```

5241 \renewenvironment{theglossary}%
5242   {\tablehead{}\tabletail{}}%
5243   \begin{supertabular}{|l>{\raggedright}p{\glsgdescwidth}%
5244     >{\raggedright}p{\glspagelistwidth}}}%
5245   {\end{supertabular}}%

```

Do nothing at the start of the table:

```
5246 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
5247 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```

5248 \renewcommand*{\glossaryentryfield}[5]{%
5249   \glstarget{##1}{\glstarget{##1}{##2} & ##3 & ##5\tabularnewline}%

```

Sub entries on a row (no name, description in second column, page list in last column):

```

5250 \renewcommand*{\glossarysubentryfield}[6]{%
5251   &
5252   \glssubentryitem{##2}%
5253   \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%

```

Blank row between groups:

```

5254 \renewcommand*{\glsgroupskip}{\ifglsgnোগroupskip\else & \tabularnewline\fi}%
5255 }

```

superragged3colborder The superragged3colborder style is like the superragged3col style, but with a border:

```
5256 \newglossarystyle{superragged3colborder}{%
```

Base it on the glostylesuperragged3col style:

```
5257 \glossarystyle{superragged3col}%
```

Put the glossary in a supertabular environment with three columns and a horizontal line in the head and tail:

```
5258 \renewenvironment{theglossary}%
5259   {\tablehead{\hline}\tabletail{\hline}%
5260    \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}|}%
5261     >{\raggedright}p{\glspagelistwidth}|}}%
5262   {\end{supertabular}}%
5263 }
```

superragged3colheader The superragged3colheader style is like the superragged3col style but with a header row:

```
5264 \newglossarystyle{superragged3colheader}{%
```

Base it on the glostylesuperragged3col style:

```
5265 \glossarystyle{superragged3col}%
```

Put the glossary in a supertabular environment with three columns, a header and no tail:

```
5266 \renewenvironment{theglossary}%
5267   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
5268    \bfseries\pagelistname\tabularnewline}\tabletail{}}%
5269   \begin{supertabular}{|l>{\raggedright}p{\glsgdescwidth}%
5270    >{\raggedright}p{\glspagelistwidth}|}%
5271   {\end{supertabular}}%
5272 }
```

superragged3colheaderborder The superragged3colheaderborder style is like the superragged3col style but with a header and border:

```
5273 \newglossarystyle{superragged3colheaderborder}{%
```

Base it on the glostylesuperragged3colborder style:

```
5274 \glossarystyle{superragged3colborder}%
```

Put the glossary in a supertabular environment with three columns, a header with horizontal lines and a horizontal line in the tail:

```
5275 \renewenvironment{theglossary}%
5276   {\tablehead{\hline
5277    \bfseries\entryname&\bfseries\descriptionname&
5278    \bfseries\pagelistname\tabularnewline\hline}%
5279   \tabletail{\hline}%
5280   \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}|}%
5281    >{\raggedright}p{\glspagelistwidth}|}}%
5282   {\end{supertabular}}%
5283 }
```

altsuperragged4col The altsuperragged4col glossary style is like altsuper4col style in the package but uses ragged right formatting in the description and page list columns.

```
5284 \newglossarystyle{altsuperragged4col}{%
```

Put the glossary in a supertabular environment with four columns and no head or tail:

```
5285 \renewenvironment{theglossary}%
5286   {\tablehead{}\tabletail{}}%
5287   \begin{supertabular}{1>{\raggedright}p{\glsgdescwidth}1%
5288     >{\raggedright}p{\glspagelistwidth}}}%
5289   {\end{supertabular}}%
```

Do nothing at the start of the table:

```
5290 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
5291 \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:

```
5292 \renewcommand*{\glossaryentryfield}[5]{%
5293   \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:

```
5294 \renewcommand*{\glossarysubentryfield}[6]{%
5295   &
5296   \glssubentryitem{##2}%
5297   \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%

```

Blank row between groups:

```
5298 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & & \tabularnewline\fi}%
5299 }
```

`\perragged4colheader` The `altsuperragged4colheader` style is like the `altsuperragged4col` style but with a header row.

```
5300 \newglossarystyle{altsuperragged4colheader}{%
```

Base it on the `glostylealtsuperragged4col` style:

```
5301 \glossarystyle{altsuperragged4col}%
```

Put the glossary in a supertabular environment with four columns, a header and no tail:

```
5302 \renewenvironment{theglossary}%
5303   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
5304     \bfseries\symbolname &
5305     \bfseries\pagelistname\tabularnewline}\tabletail{}}%
5306   \begin{supertabular}{1>{\raggedright}p{\glsgdescwidth}1%
5307     >{\raggedright}p{\glspagelistwidth}}}%
5308   {\end{supertabular}}%
5309 }
```

`\perragged4colborder` The `altsuperragged4colborder` style is like the `altsuperragged4col` style but with a border.

```
5310 \newglossarystyle{altsuperragged4colborder}{%
```

Base it on the `glostylealtsuperragged4col` style:

```
5311 \glossarystyle{altsuper4col}%
```

Put the glossary in a `supertabular` environment with four columns and a horizontal line in the head and tail:

```
5312 \renewenvironment{theglossary}%
5313   {\tablehead{\hline}\tabletail{\hline}%
5314    \begin{supertabular}%
5315      {|l|>{\raggedright}p{\glsgdescwidth}|l|}%
5316      >{\raggedright}p{\glspagelistwidth}|}}%
5317   {\end{supertabular}}%
5318 }
```

`ged4colheaderborder` The `altsuperragged4colheaderborder` style is like the `altsuperragged4col` style but with a header and border.

```
5319 \newglossarystyle{altsuperragged4colheaderborder}{%
```

Base it on the `glostylealtsuperragged4col` style:

```
5320 \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:

```
5321 \renewenvironment{theglossary}%
5322   {\tablehead{\hline
5323     \bfseries\entryname &
5324     \bfseries\descriptionname &
5325     \bfseries\symbolname &
5326     \bfseries\pagelistname\tabularnewline\hline}%
5327    \tabletail{\hline}%
5328    \begin{supertabular}%
5329      {|l|>{\raggedright}p{\glsgdescwidth}|l|}%
5330      >{\raggedright}p{\glspagelistwidth}|}}%
5331   {\end{supertabular}}%
5332 }
```

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.

```
5333 \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.

```
5334 \newglossarystyle{index}{%
```

Set the paragraph indentation and skip and define `\item` to be the same as that used by `theindex`:


```

5335 \renewenvironment{theglossary}%
5336   {\setlength{\parindent}{0pt}%
5337    \setlength{\parskip}{0pt plus 0.3pt}%
5338    \let\item\@idxitem}%
5339   {}%

```

Do nothing at the start of the environment:

```

5340 \renewcommand*{\glossaryheader}{}%

```

No group headers:

```

5341 \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.

```

5342 \renewcommand*{\glossaryentryfield}[5]{%
5343 \item\glstentryitem{##1}\textbf{\glstarget{##1}{##2}}%
5344 \ifx\relax##4\relax
5345 \else
5346   \space{##4}%
5347 \fi
5348 \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.

```

5349 \renewcommand*{\glossarysubentryfield}[6]{%
5350   \ifcase##1\relax
5351     % level 0
5352     \item
5353   \or
5354     % level 1
5355     \subitem
5356     \glssubentryitem{##2}%
5357   \else
5358     % all other levels
5359     \subsubitem
5360   \fi
5361   \textbf{\glstarget{##2}{##3}}%
5362   \ifx\relax##5\relax
5363   \else
5364     \space{##5}%
5365   \fi
5366   \space##4\glspostdescription\space ##6}%

```

Vertical gap between groups is the same as that used by indices:

```

5367 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}}

```

`indexgroup` The `indexgroup` style is like the `index` style but has headings.

```

5368 \newglossarystyle{indexgroup}{%

```

Base it on the `glostyleindex` style:

```
5369 \glossarystyle{index}%
```

Add a heading for each group. This puts the group's title in bold followed by a vertical gap.

```
5370 \renewcommand*{\glsgroupheading}[1]{%
5371   \item\textbf{\glsgetgrouptitle{##1}}\indexspace}%
5372 }
```

`indexhypergroup` The `indexhypergroup` style is like the `indexgroup` style but has hyper navigation.

```
5373 \newglossarystyle{indexhypergroup}{%
```

Base it on the `glostyleindex` style:

```
5374 \glossarystyle{index}%
```

Put navigation links to the groups at the start of the glossary:

```
5375 \renewcommand*{\glossaryheader}{%
5376   \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.

```
5377 \renewcommand*{\glsgroupheading}[1]{%
5378   \item\textbf{\glsnavhypertarget{##1}}{\glsgetgrouptitle{##1}}}%
5379   \indexspace}%
5380 }
```

`tree` The `tree` glossary style is similar in style to the `index` style, but can have arbitrary levels.

```
5381 \newglossarystyle{tree}{%
```

Set the paragraph indentation and skip:

```
5382 \renewenvironment{theglossary}%
5383   {\setlength{\parindent}{0pt}%
5384    \setlength{\parskip}{0pt plus 0.3pt}}%
5385   {}%
```

Do nothing at the start of the `theglossary` environment:

```
5386 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
5387 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries: name in bold, followed by symbol in brackets (if it exists), the description and the page list:

```
5388 \renewcommand{\glossaryentryfield}[5]{%
5389   \hangindent0pt\relax
5390   \parindent0pt\relax
5391   \glsentryitem{##1}\textbf{\glstarget{##1}{##2}}%
5392   \ifx\relax##4\relax
5393   \else
5394     \space{##4}%
5395   \fi
5396   \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.

```

5397 \renewcommand{\glossarysubentryfield}[6]{%
5398   \hangindent##1\glstreeindent\relax
5399   \parindent##1\glstreeindent\relax
5400   \ifnum##1=1\relax
5401     \glssubentryitem{##2}%
5402   \fi
5403   \textbf{\glstarget{##2}{##3}}%
5404   \ifx\relax##5\relax
5405   \else
5406     \space{##5}%
5407   \fi
5408   \space##4\glspostdescription\space ##6\par}%

```

Vertical gap between groups is the same as that used by indices:

```

5409 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}}

```

treegroup Like the tree style but the glossary groups have headings.

```

5410 \newglossarystyle{treegroup}{%

```

Base it on the `glostyletree` style:

```

5411 \glossarystyle{tree}%

```

Each group has a heading (in bold) followed by a vertical gap):

```

5412 \renewcommand{\glsgroupheading}[1]{\par
5413   \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
5414 }

```

treehypergroup The `treehypergroup` style is like the `treegroup` style, but has a set of links to the groups at the start of the glossary.

```

5415 \newglossarystyle{treehypergroup}{%

```

Base it on the `glostyletree` style:

```

5416 \glossarystyle{tree}%

```

Put navigation links to the groups at the start of the `theglossary` environment:

```

5417 \renewcommand*{\glossaryheader}{%
5418   \par\noindent\textbf{\glsnavigation}\par\indexspace}%

```

Each group has a heading (in bold with a target) followed by a vertical gap):

```

5419 \renewcommand*{\glsgroupheading}[1]{%
5420   \par\noindent
5421   \textbf{\glsnavigationtarget{##1}{\glsgrouptitle{##1}}}\par
5422   \indexspace}%
5423 }

```

\glstreeindent Length governing left indent for each level of the tree style.

```

5424 \newlength\glstreeindent
5425 \setlength{\glstreeindent}{10pt}

```

treenoname The **treenoname** glossary style is like the **tree** style, but doesn't print the name or symbol for sub-levels.

```
5426 \newglossarystyle{treenoname}{%
```

Set the paragraph indentation and skip:

```
5427 \renewenvironment{theglossary}%
5428 {\setlength{\parindent}{0pt}%
5429 \setlength{\parskip}{0pt plus 0.3pt}}%
5430 {}%
```

No header:

```
5431 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
5432 \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.

```
5433 \renewcommand{\glossaryentryfield}[5]{%
5434 \hangindent0pt\relax
5435 \parindent0pt\relax
5436 \glsentryitem{##1}\textbf{\glstarget{##1}{##2}}%
5437 \ifx\relax##4\relax
5438 \else
5439 \space{##4}%
5440 \fi
5441 \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.

```
5442 \renewcommand{\glossarysubentryfield}[6]{%
5443 \hangindent##1\glstreeindent\relax
5444 \parindent##1\glstreeindent\relax
5445 \ifnum##1=1\relax
5446 \glssubentryitem{##2}%
5447 \fi
5448 \glstarget{##2}{\strut}%
5449 ##4\glspostdescription\space ##6\par}%
```

Vertical gap between groups is the same as that used by indices:

```
5450 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
5451 }
```

treenonamegroup Like the **treenoname** style but the glossary groups have headings.

```
5452 \newglossarystyle{treenonamegroup}{%
```

Base it on the `glostyletreenoname` style:

```
5453 \glossarystyle{treenoname}%
```

Give each group a heading:

```
5454 \renewcommand{\glsgroupheading}[1]{\par
5455 \noindent\textbf{\glsgetgrouptitle{##1}}\par\indexspace}%
5456 }
```

`treenonamehypergroup` The `treenonamehypergroup` style is like the `treenonamegroup` style, but has a set of links to the groups at the start of the glossary.

```
5457 \newglossarystyle{treenonamehypergroup}{%
```

Base it on the `glostyletreenoname` style:

```
5458 \glossarystyle{treenoname}%
```

Put navigation links to the groups at the start of the `theglossary` environment:

```
5459 \renewcommand*{\glossaryheader}{%
```

```
5460 \par\noindent\textbf{\glstravigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
5461 \renewcommand*{\glsgroupheading}[1]{%
```

```
5462 \par\noindent
```

```
5463 \textbf{\glsnavigationhypertarget{##1}{\glsgroupheading{##1}}}\par
```

```
5464 \indexspace}%
```

```
5465 }
```

`\glsssetwidest` `\glsssetwidest[<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.

```
5466 \newcommand*{\glsssetwidest}[2][0]{%
```

```
5467 \expandafter\def\csname @glswidestname\romannumeral#1\endcsname{%
```

```
5468 #2}%
```

```
5469 }
```

`\@glswidestname` Initialise `\@glswidestname`.

```
5470 \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 `\glsssetwidest`.

```
5471 \newglossarystyle{alttree}{%
```

Redefine the `theglossary` environment.

```
5472 \renewenvironment{theglossary}%
```

```
5473 {\def\@gls@prevlevel{-1}%
```

```
5474 \mbox{}\par}%
```

```
5475 {\par}%
```

Set the header and group headers to nothing.

```
5476 \renewcommand*{\glossaryheader}{}%
```

```
5477 \renewcommand*{\glsgroupheading}[1]{}%
```

Redefine the way that the level 0 entries are displayed.

```
5478 \renewcommand{\glossaryentryfield}[5]{%
```

If the level hasn't changed, keep the same settings, otherwise change `\glstreeindent` accordingly.

```
5479 \ifnum\@gls@prevlevel=0\relax
```

```
5480 \else
```

Find out how big the indentation should be by measuring the widest entry.

```

5481      \settowidth{\glstreeindent}{\textbf{\@glswidestname\space}}%

```

Set the hangindent and paragraph indent.

```

5482      \hangindent\glstreeindent
5483      \parindent\glstreeindent
5484      \fi

```

Put the name to the left of the paragraph block.

```

5485      \makebox[0pt][r]{\makebox[\glstreeindent][l]{%
5486        \glstryitem{##1}\textbf{\glstarget{##1}{##2}}}%

```

If the symbol is missing, ignore it, otherwise put it in brackets.

```

5487      \ifx\relax##4\relax
5488      \else
5489        (##4)\space
5490      \fi

```

Do the description followed by the description terminator and location list.

```

5491      ##3\glspostdescription \space ##5\par

```

Set the previous level to 0.

```

5492      \def\@gls@prevlevel{0}%
5493      }%

```

Redefine the way sub-entries are displayed.

```

5494      \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.

```

5495      \ifnum##1=1\relax
5496        \glssubentryitem{##2}%
5497      \fi

```

If the level hasn't changed, keep the same settings, otherwise adjust \glstreeindent accordingly.

```

5498      \ifnum\@gls@prevlevel=##1\relax
5499      \else

```

Compute the widest entry for this level, or for level 0 if not defined for this level.

Store in \gls@tmplen

```

5500      \@ifundefined{\@glswidestname\romannumeral##1}{%
5501        \settowidth{\gls@tmplen}{\textbf{\@glswidestname\space}}{%
5502        \settowidth{\gls@tmplen}{\textbf{%
5503          \csname \@glswidestname\romannumeral##1\endcsname\space}}}%

```

Determine if going up or down a level

```

5504      \ifnum\@gls@prevlevel<##1\relax

```

Depth has increased, so add the width of the widest entry to \glstreeindent.

```

5505        \setlength\glstreeindent\gls@tmplen
5506        \addtolength\glstreeindent\parindent
5507        \parindent\glstreeindent
5508      \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`.

```

5509      \ifundefined{@glswidestname\romannumeral\@gls@prevlevel}{%
5510      \settowidth{\glstreeindent}{\textbf{%
5511      \@glswidestname\space}}}{%
5512      \settowidth{\glstreeindent}{\textbf{%
5513      \csname @glswidestname\romannumeral\@gls@prevlevel
5514      \endcsname\space}}}{%

```

Subtract this length from the previous level's paragraph indent and set to `\glstreeindent`.

```

5515      \addtolength\parindent{-\glstreeindent}%
5516      \setlength\glstreeindent\parindent
5517      \fi
5518      \fi

```

Set the hanging indentation.

```

5519      \hangindent\glstreeindent

```

Put the name to the left of the paragraph block

```

5520      \makebox[0pt][r]{\makebox[\@gls@tmplen][l]{%
5521      \textbf{\glstarget{##2}{##3}}}}}%

```

If the symbol is missing, ignore it, otherwise put it in brackets.

```

5522      \ifx##5\relax\relax
5523      \else
5524      (##5)\space
5525      \fi

```

Do the description followed by the description terminator and location list.

```

5526      ##4\glspostdescription\space ##6\par

```

Set the previous level macro to the current level.

```

5527      \def\@gls@prevlevel{##1}%
5528      }%

```

Vertical gap between groups is the same as that used by indices:

```

5529      \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
5530 }

```

almtreegroup Like the `almtree` style but the glossary groups have headings.

```

5531 \newglossarystyle{almtreegroup}{%

```

Base it on the `glostylealmtree` style:

```

5532 \glossarystyle{almtree}%

```

Give each group a heading.

```

5533 \renewcommand{\glsgroupheading}[1]{\par
5534 \def\@gls@prevlevel{-1}%
5535 \hangindent0pt\relax
5536 \parindent0pt\relax

```

```

5537 \textbf{\glsgetgrouptitle{##1}}\par\indexspace}%
5538 }

```

`almtreehypergroup` The `almtreehypergroup` style is like the `almtreegroup` style, but has a set of links to the groups at the start of the glossary.

```

5539 \newglossarystyle{almtreehypergroup}{%

```

Base it on the `glostylealmtree` style:

```

5540 \glossarystyle{almtree}%

```

Put the navigation links in the header

```

5541 \renewcommand*{\glossaryheader}{%
5542 \par
5543 \def\@gls@prevlevel{-1}%
5544 \hangindent0pt\relax
5545 \parindent0pt\relax
5546 \textbf{\glsnavigation}\par\indexspace}%

```

Put a hypertarget at the start of each group

```

5547 \renewcommand*{\glsgroupheading}[1]{%
5548 \par
5549 \def\@gls@prevlevel{-1}%
5550 \hangindent0pt\relax
5551 \parindent0pt\relax
5552 \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
5553 \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.

```

5554 \NeedsTeXFormat{LaTeX2e}
5555 \ProvidesPackage{glossaries-compatible-207}[2011/04/02 v1.0 (NLCT)]

```

`\GlsAddXdyAttribute` Adds an attribute in old format.

```

5556 \ifglsxindy
5557 \renewcommand*\GlsAddXdyAttribute[1]{%
5558 \edef\@xdyattributes{\@xdyattributes ^^J \string"#1\string"}%
5559 \expandafter\toks@\expandafter{\@xdylocref}%
5560 \edef\@xdylocref{\the\toks@ ^^J%
5561 (markup-locref
5562 :open \string"\string~n\string\setentrycounter
5563 {\noexpand\glscounter}%
5564 \expandafter\string\csname#1\endcsname
5565 \expandafter@gobble\string\{\string" ^^J
5566 :close \string"\expandafter@gobble\string\}\string" ^^J
5567 :attr \string"#1\string")}}

```


Only has an effect before \writeist:

5568 \fi

\GlsAddXdyCounters

```
5569 \renewcommand*\GlsAddXdyCounters[1]{%
5570   \GlossariesWarning{\string\GlsAddXdyCounters\space not available
5571     in compatibility mode.}%
5572 }
```

Add predefined attributes

```
5573 \GlsAddXdyAttribute{glsnumberformat}
5574 \GlsAddXdyAttribute{textrm}
5575 \GlsAddXdyAttribute{textsf}
5576 \GlsAddXdyAttribute{texttt}
5577 \GlsAddXdyAttribute{textbf}
5578 \GlsAddXdyAttribute{textmd}
5579 \GlsAddXdyAttribute{textit}
5580 \GlsAddXdyAttribute{textup}
5581 \GlsAddXdyAttribute{textsl}
5582 \GlsAddXdyAttribute{textsc}
5583 \GlsAddXdyAttribute{emph}
5584 \GlsAddXdyAttribute{glshypernumber}
5585 \GlsAddXdyAttribute{hyperrrm}
5586 \GlsAddXdyAttribute{hypersf}
5587 \GlsAddXdyAttribute{hypertt}
5588 \GlsAddXdyAttribute{hyperbf}
5589 \GlsAddXdyAttribute{hypermd}
5590 \GlsAddXdyAttribute{hyperit}
5591 \GlsAddXdyAttribute{hyperup}
5592 \GlsAddXdyAttribute{hypersl}
5593 \GlsAddXdyAttribute{hypersc}
5594 \GlsAddXdyAttribute{hyperemph}
```

\GlsAddXdyLocation Restore v2.07 definition:

```
5595 \ifglxindy
5596   \renewcommand*\GlsAddXdyLocation[2]{%
5597     \edef\@xdyuserlocationdefs{%
5598       \@xdyuserlocationdefs ^^J%
5599       (define-location-class \string"#1\string"^^J\space\space
5600         \space(#2))
5601     }%
5602     \edef\@xdyuserlocationnames{%
5603       \@xdyuserlocationnames^^J\space\space\space
5604       \string"#1\string"%
5605     }
5606 \fi
```

\@do@wrglossary

```
5607 \renewcommand*\@do@wrglossary[1]{%
```

Determine whether to use xindy or makeindex syntax

```
5608 \ifglxsindy
```

Need to determine if the formatting information starts with a (or) indicating a range.

```
5609 \expandafter\@glo@check@mkidxrangechar\@glsnumberformat\@nil
5610 \def\@glo@range{}%
5611 \expandafter\if\@glo@prefix(\relax
5612 \def\@glo@range{:open-range}%
5613 \else
5614 \expandafter\if\@glo@prefix)\relax
5615 \def\@glo@range{:close-range}%
5616 \fi
5617 \fi
```

Get the location and escape any special characters

```
5618 \protected@edef\@glslocref{\theglentrycounter}%
5619 \@gls@checkmkidxchars\@glslocref
```

Write to the glossary file using xindy syntax.

```
5620 \glossary[\csname glo@#1@type\endcsname]{%
5621 (indexentry :tkey (\csname glo@#1@index\endcsname)
5622 :locref \string"\@glslocref\string" %
5623 :attr \string"\@glo@suffix\string" \@glo@range
5624 )
5625 }%
5626 \else
```

Convert the format information into the format required for makeindex

```
5627 \@set@glo@numformat\@glo@numfmt\@gls@counter\@glsnumberformat
```

Write to the glossary file using makeindex syntax.

```
5628 \glossary[\csname glo@#1@type\endcsname]{%
5629 \string\glossaryentry{\csname glo@#1@index\endcsname
5630 \@gls@encapchar\@glo@numfmt}{\theglentrycounter}}%
5631 \fi
5632 }
```

\@set@glo@numformat Only had 3 arguments in v2.07

```
5633 \def\@set@glo@numformat#1#2#3{%
5634 \expandafter\@glo@check@mkidxrangechar#3\@nil
5635 \protected@edef#1{%
5636 \@glo@prefix setentrycounter[] {#2}%
5637 \expandafter\string\csname\@glo@suffix\endcsname
5638 }%
5639 \@gls@checkmkidxchars#1%
5640 }
```

\writeist Redefine \writeist back to the way it was in v2.07, but change \istfile to \glswrite.

```

5641 \ifglxindy
5642   \def\writeist{%
5643     \openout\glswrite=\istfilename
5644     \write\glswrite{;; xindy style file created by the glossaries
5645       package in compatible-2.07 mode}%
5646     \write\glswrite{;; for document '\jobname' on
5647       \the\year-\the\month-\the\day}%
5648     \write\glswrite{^^J; required styles^^J}
5649     \@for\@xdystyle:=\@xdyrequiredstyles\do{%
5650       \ifx\@xdystyle\@empty
5651       \else
5652         \protected@write\glswrite{{(require
5653           \string"\@xdystyle.xdy\string")}}%
5654       \fi
5655     }%
5656     \write\glswrite{^^J%
5657       ; list of allowed attributes (number formats)^^J}%
5658     \write\glswrite{(define-attributes ((\@xdyattributes)))}%
5659     \write\glswrite{^^J; user defined alphabets^^J}%
5660     \write\glswrite{\@xdyuseralphabets}%
5661     \write\glswrite{^^J; location class definitions^^J}%
5662     \protected@edef\@gls@roman{\@roman{0}\string"
5663       \string"roman-numbers-lowercase\string" :sep \string"}}%
5664     \@onelevel@sanitize\@gls@roman
5665     \edef\@tmp{\string" \string"roman-numbers-lowercase\string"
5666       :sep \string"}%
5667     \@onelevel@sanitize\@tmp
5668     \ifx\@tmp\@gls@roman
5669       \write\glswrite{(define-location-class
5670         \string"roman-page-numbers\string"^^J\space\space\space
5671         (\string"roman-numbers-lowercase\string")
5672         :min-range-length \@glsminrange)}}%
5673     \else
5674       \write\glswrite{(define-location-class
5675         \string"roman-page-numbers\string"^^J\space\space\space
5676         (:sep "\@gls@roman")
5677         :min-range-length \@glsminrange)}}%
5678     \fi
5679     \write\glswrite{(define-location-class
5680       \string"Roman-page-numbers\string"^^J\space\space\space
5681       (\string"roman-numbers-uppercase\string")
5682       :min-range-length \@glsminrange)}}%
5683     \write\glswrite{(define-location-class
5684       \string"arabic-page-numbers\string"^^J\space\space\space
5685       (\string"arabic-numbers\string")
5686       :min-range-length \@glsminrange)}}%
5687     \write\glswrite{(define-location-class
5688       \string"alpha-page-numbers\string"^^J\space\space\space
5689       (\string"alpha\string")

```

```

5690         :min-range-length \@glsminrange))}%
5691 \write\glswrite{(define-location-class
5692   \string"Alpha-page-numbers\string"^^J\space\space\space
5693   (\string"ALPHA\string")
5694   :min-range-length \@glsminrange))}%
5695 \write\glswrite{(define-location-class
5696   \string"Appendix-page-numbers\string"^^J\space\space\space
5697   (\string"ALPHA\string"
5698   :sep \string"\@glsAlphacompositor\string"
5699   \string"arabic-numbers\string")
5700   :min-range-length \@glsminrange))}%
5701 \write\glswrite{(define-location-class
5702   \string"arabic-section-numbers\string"^^J\space\space\space
5703   (\string"arabic-numbers\string"
5704   :sep \string"\glscompositor\string"
5705   \string"arabic-numbers\string")
5706   :min-range-length \@glsminrange))}%
5707 \write\glswrite{^^J; user defined location classes}%
5708 \write\glswrite{@xdyuserlocationdefs}%
5709 \write\glswrite{^^J; define cross-reference class^^J}%
5710 \write\glswrite{(define-crossref-class \string"see\string"
5711   :unverified )}%
5712 \write\glswrite{(markup-crossref-list
5713   :class \string"see\string"^^J\space\space\space
5714   :open \string"\string\glsseeformat\string"
5715   :close \string"{}\string"))}%
5716 \write\glswrite{^^J; define the order of the location classes}%
5717 \write\glswrite{(define-location-class-order
5718   (\@xdylocationclassorder))}%
5719 \write\glswrite{^^J; define the glossary markup^^J}%
5720 \write\glswrite{(markup-index^^J\space\space\space
5721   :open \string"\string
5722   \glossarysection[\string\glossarytoctitle]{\string
5723   \glossarytitle}\string\glossarypreamble\string~n\string\begin
5724   {theglossary}\string\glossaryheader\string~n\string" ^^J\space
5725   \space\space:close \string"\expandafter\@gobble
5726   \string%\string~n\string
5727   \end{theglossary}\string\glossarypostamble
5728   \string~n\string" ^^J\space\space\space
5729   :tree)}}%
5730 \write\glswrite{(markup-letter-group-list
5731   :sep \string"\string\glsgroupskip\string~n\string"))}%
5732 \write\glswrite{(markup-indexentry
5733   :open \string"\string\relax \string\glsresetentrylist
5734   \string~n\string"))}%
5735 \write\glswrite{(markup-locclass-list :open
5736   \string"\glsopenbrace\string\glossaryentrynumbers
5737   \glsopenbrace\string\relax\space \string"^^J\space\space\space
5738   :sep \string", \string"

```

```

5739 :close \string"\glsclosebrace\glsclosebrace\string"))}%
5740 \write\glswrite{(markup-locoref-list
5741 :sep \string"\string\delimN\space\string"))}%
5742 \write\glswrite{(markup-range
5743 :sep \string"\string\delimR\space\string"))}%
5744 \@onelevel@sanitize\gls@suffixF
5745 \@onelevel@sanitize\gls@suffixFF
5746 \ifx\gls@suffixF\@empty
5747 \else
5748 \write\glswrite{(markup-range
5749 :close "\gls@suffixF" :length 1 :ignore-end))}%
5750 \fi
5751 \ifx\gls@suffixFF\@empty
5752 \else
5753 \write\glswrite{(markup-range
5754 :close "\gls@suffixFF" :length 2 :ignore-end))}%
5755 \fi
5756 \write\glswrite{^^J; define format to use for locations^^J}%
5757 \write\glswrite{@xdylocoref}%
5758 \write\glswrite{^^J; define letter group list format^^J}%
5759 \write\glswrite{(markup-letter-group-list
5760 :sep \string"\string\glsgroupskip\string~n\string"))}%
5761 \write\glswrite{^^J; letter group headings^^J}%
5762 \write\glswrite{(markup-letter-group
5763 :open-head \string"\string\glsgroupheading
5764 \glsopenbrace\string"^^J\space\space\space
5765 :close-head \string"\glsclosebrace\string"))}%
5766 \write\glswrite{^^J; additional letter groups^^J}%
5767 \write\glswrite{@xdylettergroups}%
5768 \write\glswrite{^^J; additional sort rules^^J}%
5769 \write\glswrite{@xdysortrules}%
5770 \noist}
5771 \else
5772 \edef\@gls@actualchar{\string?}
5773 \edef\@gls@encapchar{\string|}
5774 \edef\@gls@levelchar{\string!}
5775 \edef\@gls@quotechar{\string"}
5776 \def\writeist{\relax
5777 \openout\glswrite=\istfilename
5778 \write\glswrite{\expandafter\@gobble\string\% makeindex style file
5779 created by the glossaries package}
5780 \write\glswrite{\expandafter\@gobble\string\% for document
5781 '\jobname' on \the\year-\the\month-\the\day}
5782 \write\glswrite{actual '\@gls@actualchar'}
5783 \write\glswrite{encap '\@gls@encapchar'}
5784 \write\glswrite{level '\@gls@levelchar'}
5785 \write\glswrite{quote '\@gls@quotechar'}
5786 \write\glswrite{keyword \string"\string\glossaryentry\string"}
5787 \write\glswrite{preamble \string"\string\glossarysection[\string

```

```

5788     \glossarytoctitle]{\string\glossarytitle}\string
5789     \glossarypreamble\string\n\string\begin{theglossary}\string
5790     \glossaryheader\string\n\string"}
5791 \write\glswrite{postamble \string"\string%\string\n\string
5792     \end{theglossary}\string\glossarypostamble\string\n
5793     \string"}
5794 \write\glswrite{group_skip \string"\string\glsgroupskip\string\n
5795     \string"}
5796 \write\glswrite{item_0 \string"\string%\string\n\string"}
5797 \write\glswrite{item_1 \string"\string%\string\n\string"}
5798 \write\glswrite{item_2 \string"\string%\string\n\string"}
5799 \write\glswrite{item_01 \string"\string%\string\n\string"}
5800 \write\glswrite{item_x1
5801     \string"\string\relax \string\glsresetentrylist\string\n
5802     \string"}
5803 \write\glswrite{item_12 \string"\string%\string\n\string"}
5804 \write\glswrite{item_x2
5805     \string"\string\relax \string\glsresetentrylist\string\n
5806     \string"}
5807 \write\glswrite{delim_0 \string"\string\{\string
5808     \glossaryentrynumbers\string\{\string\relax \string"}
5809 \write\glswrite{delim_1 \string"\string\{\string
5810     \glossaryentrynumbers\string\{\string\relax \string"}
5811 \write\glswrite{delim_2 \string"\string\{\string
5812     \glossaryentrynumbers\string\{\string\relax \string"}
5813 \write\glswrite{delim_t \string"\string\}\string\}\string"}
5814 \write\glswrite{delim_n \string"\string\delimN \string"}
5815 \write\glswrite{delim_r \string"\string\delimR \string"}
5816 \write\glswrite{headings_flag 1}
5817 \write\glswrite{heading_prefix
5818     \string"\string\glsgroupheading\string\{\string"}
5819 \write\glswrite{heading_suffix
5820     \string"\string\}\string\relax
5821     \string\glsresetentrylist \string"}
5822 \write\glswrite{symhead_positive \string"glssymbols\string"}
5823 \write\glswrite{numhead_positive \string"glssymbols\string"}
5824 \write\glswrite{page_compositor \string"glscpositor\string"}
5825 \@gls@escbsdq\gls@suffixF
5826 \@gls@escbsdq\gls@suffixFF
5827 \ifx\gls@suffixF\@empty
5828 \else
5829     \write\glswrite{suffix_2p \string"\gls@suffixF\string"}
5830 \fi
5831 \ifx\gls@suffixFF\@empty
5832 \else
5833     \write\glswrite{suffix_3p \string"\gls@suffixFF\string"}
5834 \fi
5835 \noist
5836 }

```

```
5837 \fi
```

```
\noist
```

```
5838 \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.

```
5839 \NeedsTeXFormat{LaTeX2e}
```

Package version number now in line with main glossaries package number but will only be updated when `glossaries-accsupp.sty` is modified.

```
5840 \ProvidesPackage{glossaries-accsupp}[2011/04/02 v3.0 (NLCT)]
```

```
5841 Experimental glossaries accessibility
```

Pass all options to glossaries:

```
5842 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{glossaries}}
```

Process options:

```
5843 \ProcessOptions
```

Required packages:

```
5844 \RequirePackage{glossaries}
```

```
5845 \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:

```
5846 \define@key{glossentry}{access}{%
```

```
5847 \def\@glo@access{#1}%
```

```
5848 }
```

textaccess The replacement text corresponding to the text key:

```
5849 \define@key{glossentry}{textaccess}{%
```

```
5850 \def\@glo@textaccess{#1}%
```

```
5851 }
```

firstaccess The replacement text corresponding to the first key:

```
5852 \define@key{glossentry}{firstaccess}{%
```

```
5853 \def\@glo@firstaccess{#1}%
```

```
5854 }
```

pluralaccess The replacement text corresponding to the plural key:

```
5855 \define@key{glossentry}{pluralaccess}{%
5856   \def\@glo@pluralaccess{#1}%
5857 }
```

firstpluralaccess The replacement text corresponding to the firstplural key:

```
5858 \define@key{glossentry}{firstpluralaccess}{%
5859   \def\@glo@firstpluralaccess{#1}%
5860 }
```

symbolaccess The replacement text corresponding to the symbol key:

```
5861 \define@key{glossentry}{symbolaccess}{%
5862   \def\@glo@symbolaccess{#1}%
5863 }
```

symbolpluralaccess The replacement text corresponding to the symbolplural key:

```
5864 \define@key{glossentry}{symbolpluralaccess}{%
5865   \def\@glo@symbolpluralaccess{#1}%
5866 }
```

descriptionaccess The replacement text corresponding to the description key:

```
5867 \define@key{glossentry}{descriptionaccess}{%
5868   \def\@glo@descaccess{#1}%
5869 }
```

descriptionpluralaccess The replacement text corresponding to the descriptionplural key:

```
5870 \define@key{glossentry}{descriptionpluralaccess}{%
5871   \def\@glo@descpluralaccess{#1}%
5872 }
```

shortaccess The replacement text corresponding to the short key:

```
5873 \define@key{glossentry}{shortaccess}{%
5874   \def\@glo@shortaccess{#1}%
5875 }
```

shortpluralaccess The replacement text corresponding to the shortplural key:

```
5876 \define@key{glossentry}{shortpluralaccess}{%
5877   \def\@glo@shortpluralaccess{#1}%
5878 }
```

longaccess The replacement text corresponding to the long key:

```
5879 \define@key{glossentry}{longaccess}{%
5880   \def\@glo@longaccess{#1}%
5881 }
```

longpluralaccess The replacement text corresponding to the longplural key:

```
5882 \define@key{glossentry}{longpluralaccess}{%
5883   \def\@glo@longpluralaccess{#1}%
5884 }
```


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.

```
5885 \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):

```
5886 \let\@gls@oldnewglossaryentryprehook\@newglossaryentryprehook
5887 \renewcommand*{\@newglossaryentryprehook}{%
5888   \@gls@oldnewglossaryentryprehook
5889   \def\@glo@access{\@glo@symbol}%
```

Initialise the other keys:

```
5890   \def\@glo@textaccess{\@glo@access}%
5891   \def\@glo@firstaccess{\@glo@access}%
5892   \def\@glo@pluralaccess{\@glo@textaccess}%
5893   \def\@glo@firstpluralaccess{\@glo@pluralaccess}%
5894   \def\@glo@symbolaccess{\relax}%
5895   \def\@glo@symbolpluralaccess{\@glo@symbolaccess}%
5896   \def\@glo@descaccess{\relax}%
5897   \def\@glo@descpluralaccess{\@glo@descaccess}%
5898   \def\@glo@shortaccess{\relax}%
5899   \def\@glo@shortpluralaccess{\@glo@shortaccess}%
5900   \def\@glo@longaccess{\relax}%
5901   \def\@glo@longpluralaccess{\@glo@longaccess}%
5902 }
```

Add to the end hook:

```
5903 \let\@gls@oldnewglossaryentryposthook\@newglossaryentryposthook
5904 \renewcommand*{\@newglossaryentryposthook}{%
5905   \@gls@oldnewglossaryentryposthook
```

Store the access information:

```
5906   \expandafter
5907     \protected@xdef\csname glo@\@glo@label @access\endcsname{%
5908       \@glo@access}%
5909   \expandafter
5910     \protected@xdef\csname glo@\@glo@label @textaccess\endcsname{%
5911       \@glo@textaccess}%
5912   \expandafter
5913     \protected@xdef\csname glo@\@glo@label @firstaccess\endcsname{%
5914       \@glo@firstaccess}%
5915   \expandafter
5916     \protected@xdef\csname glo@\@glo@label @pluralaccess\endcsname{%
5917       \@glo@pluralaccess}%
5918   \expandafter
5919     \protected@xdef\csname glo@\@glo@label @firstpluralaccess\endcsname{%
5920       \@glo@firstpluralaccess}%
5921   \expandafter
```

```

5922 \protected@xdef\csname glo@\@glo@label @symbolaccess\endcsname{%
5923 \@glo@symbolaccess}%
5924 \expandafter
5925 \protected@xdef\csname glo@\@glo@label @symbolpluralaccess\endcsname{%
5926 \@glo@symbolpluralaccess}%
5927 \expandafter
5928 \protected@xdef\csname glo@\@glo@label @descaccess\endcsname{%
5929 \@glo@descaccess}%
5930 \expandafter
5931 \protected@xdef\csname glo@\@glo@label @descpluralaccess\endcsname{%
5932 \@glo@descpluralaccess}%
5933 \expandafter
5934 \protected@xdef\csname glo@\@glo@label @shortaccess\endcsname{%
5935 \@glo@shortaccess}%
5936 \expandafter
5937 \protected@xdef\csname glo@\@glo@label @shortpluralaccess\endcsname{%
5938 \@glo@shortpluralaccess}%
5939 \expandafter
5940 \protected@xdef\csname glo@\@glo@label @longaccess\endcsname{%
5941 \@glo@longaccess}%
5942 \expandafter
5943 \protected@xdef\csname glo@\@glo@label @longpluralaccess\endcsname{%
5944 \@glo@longpluralaccess}%
5945 }

```

5.2 Accessing Replacement Text

`\glsentryaccess` Get the value of the access key for the entry with the given label:

```

5946 \newcommand*{\glsentryaccess}[1]{%
5947 \csname glo@#1@access\endcsname
5948 }

```

`\glsentrytextaccess` Get the value of the textaccess key for the entry with the given label:

```

5949 \newcommand*{\glsentrytextaccess}[1]{%
5950 \csname glo@#1@textaccess\endcsname
5951 }

```

`\glsentryfirstaccess` Get the value of the firstaccess key for the entry with the given label:

```

5952 \newcommand*{\glsentryfirstaccess}[1]{%
5953 \csname glo@#1@firstaccess\endcsname
5954 }

```

`\glsentrypluralaccess` Get the value of the pluralaccess key for the entry with the given label:

```

5955 \newcommand*{\glsentrypluralaccess}[1]{%
5956 \csname glo@#1@pluralaccess\endcsname
5957 }

```

`\glsentryfirstpluralaccess` Get the value of the firstpluralaccess key for the entry with the given label:

```

5958 \newcommand*{\glsentryfirstpluralaccess}[1]{%
5959   \csname glo@#1@firstpluralaccess\endcsname
5960 }

\glsentrysymbolaccess   Get the value of the symbolaccess key for the entry with the given label:
5961 \newcommand*{\glsentrysymbolaccess}[1]{%
5962   \csname glo@#1@symbolaccess\endcsname
5963 }

\glsentrysymbolpluralaccess   Get the value of the symbolpluralaccess key for the entry with the given label:
5964 \newcommand*{\glsentrysymbolpluralaccess}[1]{%
5965   \csname glo@#1@symbolpluralaccess\endcsname
5966 }

\glsentrydescaccess   Get the value of the descriptionaccess key for the entry with the given label:
5967 \newcommand*{\glsentrydescaccess}[1]{%
5968   \csname glo@#1@descaccess\endcsname
5969 }

\glsentrydescpluralaccess   Get the value of the descriptionpluralaccess key for the entry with the given label:
5970 \newcommand*{\glsentrydescpluralaccess}[1]{%
5971   \csname glo@#1@descaccess\endcsname
5972 }

\glsentryshortaccess   Get the value of the shortaccess key for the entry with the given label:
5973 \newcommand*{\glsentryshortaccess}[1]{%
5974   \csname glo@#1@shortaccess\endcsname
5975 }

\glsentryshortpluralaccess   Get the value of the shortpluralaccess key for the entry with the given label:
5976 \newcommand*{\glsentryshortpluralaccess}[1]{%
5977   \csname glo@#1@shortpluralaccess\endcsname
5978 }

\glsentrylongaccess   Get the value of the longaccess key for the entry with the given label:
5979 \newcommand*{\glsentrylongaccess}[1]{%
5980   \csname glo@#1@longaccess\endcsname
5981 }

\glsentrylongpluralaccess   Get the value of the longpluralaccess key for the entry with the given label:
5982 \newcommand*{\glsentrylongpluralaccess}[1]{%
5983   \csname glo@#1@longpluralaccess\endcsname
5984 }

\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.)

```
5985 \newcommand*{\glsaccsupp}[2]{%
5986   \BeginAccSupp{ActualText=#1}#2\EndAccSupp{}%
5987 }
```

`\xglsaccsupp` Fully expands replacement text before calling `\glsaccsupp`

```
5988 \newcommand*{\xglsaccsupp}[2]{%
5989   \protected@edef\@gls@replacementtext{#1}%
5990   \expandafter\xglsaccsupp\expandafter{\@gls@replacementtext}{#2}%
5991 }
```

`\glsnameaccessdisplay` Displays the first argument with the accessibility text for the entry with the label given by the second argument (if set).

```
5992 \DeclareRobustCommand*{\glsnameaccessdisplay}[2]{%
5993   \protected@edef\@glo@access{\glsentryaccess{#2}}%
5994   \ifx\@glo@access\@gls@noaccess
5995     #1%
5996   \else
5997     \xglsaccsupp{\@glo@access}{#1}%
5998   \fi
5999 }
```

`\glsentrytextaccessdisplay` As above but for the `textaccess` replacement text.

```
6000 \DeclareRobustCommand*{\glsentrytextaccessdisplay}[2]{%
6001   \protected@edef\@glo@access{\glsentrytextaccess{#2}}%
6002   \ifx\@glo@access\@gls@noaccess
6003     #1%
6004   \else
6005     \xglsaccsupp{\@glo@access}{#1}%
6006   \fi
6007 }
```

`\glsentrypluralaccessdisplay` As above but for the `pluralaccess` replacement text.

```
6008 \DeclareRobustCommand*{\glsentrypluralaccessdisplay}[2]{%
6009   \protected@edef\@glo@access{\glsentrypluralaccess{#2}}%
6010   \ifx\@glo@access\@gls@noaccess
6011     #1%
6012   \else
6013     \xglsaccsupp{\@glo@access}{#1}%
6014   \fi
6015 }
```

`\glsentryfirstaccessdisplay` As above but for the `firstaccess` replacement text.

```
6016 \DeclareRobustCommand*{\glsentryfirstaccessdisplay}[2]{%
6017   \protected@edef\@glo@access{\glsentryfirstaccess{#2}}%
6018   \ifx\@glo@access\@gls@noaccess
6019     #1%
```

```

6020 \else
6021   \xglsaccsupp{\@glo@access}{#1}%
6022 \fi
6023 }

```

pluralaccessdisplay As above but for the firstpluralaccess replacement text.

```

6024 \DeclareRobustCommand*\glsfirstpluralaccessdisplay}[2]{%
6025   \protected@edef\@glo@access{\glsentryfirstpluralaccess{#2}}%
6026   \ifx\@glo@access\@gls@noaccess
6027     #1%
6028   \else
6029     \xglsaccsupp{\@glo@access}{#1}%
6030   \fi
6031 }

```

symbolaccessdisplay As above but for the symbolaccess replacement text.

```

6032 \DeclareRobustCommand*\glsymbolaccessdisplay}[2]{%
6033   \protected@edef\@glo@access{\glsentrysymbolaccess{#2}}%
6034   \ifx\@glo@access\@gls@noaccess
6035     #1%
6036   \else
6037     \xglsaccsupp{\@glo@access}{#1}%
6038   \fi
6039 }

```

pluralaccessdisplay As above but for the symbolpluralaccess replacement text.

```

6040 \DeclareRobustCommand*\glsymbolpluralaccessdisplay}[2]{%
6041   \protected@edef\@glo@access{\glsentrysymbolpluralaccess{#2}}%
6042   \ifx\@glo@access\@gls@noaccess
6043     #1%
6044   \else
6045     \xglsaccsupp{\@glo@access}{#1}%
6046   \fi
6047 }

```

descriptionaccessdisplay As above but for the descriptionaccess replacement text.

```

6048 \DeclareRobustCommand*\glsdescriptionaccessdisplay}[2]{%
6049   \protected@edef\@glo@access{\glsentrydescaccess{#2}}%
6050   \ifx\@glo@access\@gls@noaccess
6051     #1%
6052   \else
6053     \xglsaccsupp{\@glo@access}{#1}%
6054   \fi
6055 }

```

pluralaccessdisplay As above but for the descriptionpluralaccess replacement text.

```

6056 \DeclareRobustCommand*\glsdescriptionpluralaccessdisplay}[2]{%
6057   \protected@edef\@glo@access{\glsentrydescpluralaccess{#2}}%
6058   \ifx\@glo@access\@gls@noaccess

```

```

6059    #1%
6060    \else
6061        \xglsaccsupp{\@glo@access}{#1}%
6062    \fi
6063 }

```

`sshortaccessdisplay` As above but for the `shortaccess` replacement text.

```

6064 \DeclareRobustCommand*{\glsshortaccessdisplay}[2]{%
6065     \protected@edef\@glo@access{\glstryshortaccess{#2}}%
6066     \ifx\@glo@access\@gls@noaccess
6067         #1%
6068     \else
6069         \xglsaccsupp{\@glo@access}{#1}%
6070     \fi
6071 }

```

`pluralaccessdisplay` As above but for the `shortpluralaccess` replacement text.

```

6072 \DeclareRobustCommand*{\glsshortpluralaccessdisplay}[2]{%
6073     \protected@edef\@glo@access{\glstryshortpluralaccess{#2}}%
6074     \ifx\@glo@access\@gls@noaccess
6075         #1%
6076     \else
6077         \xglsaccsupp{\@glo@access}{#1}%
6078     \fi
6079 }

```

`lslongaccessdisplay` As above but for the `longaccess` replacement text.

```

6080 \DeclareRobustCommand*{\glslongaccessdisplay}[2]{%
6081     \protected@edef\@glo@access{\glstrylongaccess{#2}}%
6082     \ifx\@glo@access\@gls@noaccess
6083         #1%
6084     \else
6085         \xglsaccsupp{\@glo@access}{#1}%
6086     \fi
6087 }

```

`pluralaccessdisplay` As above but for the `longpluralaccess` replacement text.

```

6088 \DeclareRobustCommand*{\glslongpluralaccessdisplay}[2]{%
6089     \protected@edef\@glo@access{\glstrylongpluralaccess{#2}}%
6090     \ifx\@glo@access\@gls@noaccess
6091         #1%
6092     \else
6093         \xglsaccsupp{\@glo@access}{#1}%
6094     \fi
6095 }

```

`\glsaccessdisplay` Gets the replacement text corresponding to the named key given by the first argument and calls the appropriate command defined above.

```

6096 \DeclareRobustCommand*\glsaccessdisplay}[3]{%
6097   \@ifundefined{gls#1accessdisplay}%
6098   {%
6099     \PackageError{glossaries-accsupp}{No accessibility support
6100       for key ‘#1’}{%
6101     }%
6102   {%
6103     \csname gls#1accessdisplay\endcsname{#2}{#3}%
6104   }%
6105 }

```

`\@gls@` Redefine `\@gls@` to change the way the link text is defined

```

6106 \def\@gls@#1#2[#3]{%
6107   \glsdoifexists{#2}%
6108   {%
6109     \edef\@glo@type{\glsentrytype{#2}}%
        Save options in \@gls@link@opts and label in \@gls@link@label
6110     \def\@gls@link@opts{#1}%
6111     \def\@gls@link@label{#2}%
        Determine what the link text should be (this is stored in \@glo@text). This is
        no longer expanded.
6112     \ifglsused{#2}%
6113     {%
6114       \def\@glo@text{\csname gls@\@glo@type @display\endcsname
6115         {\glsfirstaccessdisplay{\glsentryfirst{#2}}{#2}}%
6116         {\glsdescriptionaccessdisplay{\glsentrydesc{#2}}{#2}}%
6117         {\glsymbolaccessdisplay{\glsentrysymbol{#2}}{#2}}%
6118         {#3}}%
6119     }%
6120     {%
6121       \def\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
6122         {\glsfirstaccessdisplay{\glsentryfirst{#2}}{#2}}%
6123         {\glsdescriptionaccessdisplay{\glsentrydesc{#2}}{#2}}%
6124         {\glsymbolaccessdisplay{\glsentrysymbol{#2}}{#2}}%
6125         {#3}}%
6126     }%
        Call \@gls@link. If footnote package option has been used, suppress hyperlink
        for first use.
6127     \ifglsused{#2}%
6128     {%
6129       \@gls@link[#1]{#2}{\@glo@text}%
6130     }%
6131     {%
6132       \gls@checkisacronymlist\@glo@type
6133       \ifthenelse{\boolean{glsisacronymlist}\AND
6134         \boolean{glsacrfootnote}}{\OR\NOT\boolean{glshyperfirst}}%
6135     }%

```

```

6136      \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
6137  }%
6138  {%
6139      \@gls@link[#1]{#2}{\@glo@text}%
6140  }%
6141  }%

```

Indicate that this entry has now been used

```

6142      \glsunset{#2}%
6143  }%
6144 }

```

\@Gls@

```

6145 \def\@Gls@#1#2[#3]{%
6146   \glsdoifexists{#2}%
6147   {%
6148     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

6149     \def\@gls@link@opts{#1}%
6150     \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

```

6151     \ifglsused{#2}%
6152     {%
6153       \def\@glo@text{\csname gls@\@glo@type @display\endcsname
6154         {\glsfirstaccessdisplay{\Glsentrytext{#2}}{#2}}}%
6155       {\glsdescriptionaccessdisplay{\glsentrydesc{#2}}{#2}}%
6156       {\glsymbolaccessdisplay{\glsentrysymbol{#2}}{#2}}%
6157       {#3}}%
6158     }%
6159     {%
6160       \def\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
6161         {\glsfirstaccessdisplay{\Glsentryfirst{#2}}{#2}}%
6162         {\glsdescriptionaccessdisplay{\glsentrydesc{#2}}{#2}}%
6163         {\glsymbolaccessdisplay{\glsentrysymbol{#2}}{#2}}%
6164         {#3}}%
6165     }%

```

Call \@gls@link. If footnote package option has been used, suppress hyperlink for first use.

```

6166     \ifglsused{#2}%
6167     {%
6168       \@gls@link[#1]{#2}{\@glo@text}%
6169     }%
6170     {%
6171       \gls@checkisacronymlist\@glo@type
6172       \ifthenelse{\boolean{glsisacronymlist}}{\AND

```



```

6173     \boolean{glsacrfootnote}\) \OR\NOT\boolean{glshyperfirst}}}%
6174     {%
6175     \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
6176     }%
6177     {%
6178     \@gls@link[#1]{#2}{\@glo@text}%
6179     }%
6180 }%

```

Indicate that this entry has now been used

```

6181     \glsunset{#2}%
6182     }%
6183 }

```

\@GLS@

```

6184 \def\@GLS@#1#2[#3]{%
6185   \glsdoifexists{#2}{%
6186     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

6187   \def\@gls@link@opts{#1}%
6188   \def\@gls@link@label{#2}%

```

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

```

6189   \ifglsused{#2}%
6190   {%
6191     \def\@glo@text{\csname gls@\@glo@type @display\endcsname
6192       {\gls@textaccessdisplay{\glsentrytext{#2}}{#2}}}%
6193     {\gls@descriptionaccessdisplay{\glsentrydesc{#2}}{#2}}}%
6194     {\gls@symbolaccessdisplay{\glsentrysymbol{#2}}{#2}}}%
6195     {#3}}%
6196   }%
6197   {%
6198     \edef\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
6199       {\gls@firstaccessdisplay{\glsentryfirst{#2}}{#2}}}%
6200       {\gls@descriptionaccessdisplay{\glsentrydesc{#2}}{#2}}}%
6201       {\gls@symbolaccessdisplay{\glsentrysymbol{#2}}{#2}}}%
6202       {#3}}%
6203   }%

```

Call \@gls@link If footnote package option has been used, suppress hyperlink for first use.

```

6204   \ifglsused{#2}%
6205   {%
6206     \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}}%
6207   }%
6208   {%
6209     \gls@checkisacronymlist\@glo@type
6210     \ifthenelse{(\boolean{glsisacronymlist})\AND
6211       \boolean{glsacrfootnote}\) \OR\NOT\boolean{glshyperfirst}}{%
6212       \@gls@link[#1,hyper=false]{#2}{\MakeUppercase{\@glo@text}}}%

```

```

6213     }%
6214     {%
6215         \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
6216     }%
6217 }%

```

Indicate that this entry has now been used

```

6218     \glsunset{#2}%
6219 }%
6220 }

```

\@gls@pl@

```

6221 \def\@glspl@#1#2[#3]{%
6222     \glsdoifexists{#2}%
6223     {%
6224         \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

6225     \def\@gls@link@opts{#1}%
6226     \def\@gls@link@label{#2}%

```

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

```

6227     \ifglsused{#2}%
6228     {%
6229         \def\@glo@text{\csname gls@\@glo@type @display\endcsname
6230             {\glspluralaccessdisplay{\glsentryplural{#2}}{#2}}%
6231             {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
6232             {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
6233             {#3}}%
6234     }%
6235     {%
6236         \def\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
6237             {\glsfirstpluralaccessdisplay{\glsentryfirstplural{#2}}{#2}}%
6238             {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
6239             {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
6240             {#3}}%
6241     }%

```

Call \@gls@link If footnote package option has been used, suppress hyperlink for first use.

```

6242     \ifglsused{#2}%
6243     {%
6244         \@gls@link[#1]{#2}{\@glo@text}%
6245     }%
6246     {%
6247         \gls@checkisacronymlist\@glo@type
6248         \ifthenelse{(\boolean{@glsisacronymlist})\AND
6249             \boolean{glsacrfootnote}}{\OR\NOT\boolean{glshyperfirst}}%
6250         {%
6251             \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
6252         }%

```

```

6253      {%
6254      \@gls@link[#1]{#2}{\@glo@text}%
6255      }%
6256      }%

```

Indicate that this entry has now been used

```

6257      \glsunset{#2}%
6258      }%
6259      }

```

\@Glspl@

```

6260 \def\@Glspl@#1#2[#3]{%
6261   \glsdoifexists{#2}%
6262   {%
6263     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

6264     \def\@gls@link@opts{#1}%
6265     \def\@gls@link@label{#2}%

```

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

```

6266     \ifglsused{#2}%
6267     {%
6268       \def\@glo@text{\csname gls@\@glo@type @display\endcsname
6269         {\glspluralaccessdisplay{\Glsentryplural{#2}}{#2}}%
6270         {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
6271         {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
6272         {#3}}%
6273     }%
6274     {%
6275       \def\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
6276         {\glsfirstpluralaccessdisplay{\Glsentryfirstplural{#2}}{#2}}%
6277         {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
6278         {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
6279         {#3}}%
6280     }%

```

Call \@gls@link If footnote package option has been used, suppress hyperlink for first use.

```

6281     \ifglsused{#2}%
6282     {%
6283       \@gls@link[#1]{#2}{\@glo@text}%
6284     }%
6285     {%
6286       \ifthenelse{\equal{\@glo@type}{\acronymtype}\and
6287         \boolean{glsacrfootnote}}%
6288       {%
6289         \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
6290       }%
6291       {%
6292         \@gls@link[#1]{#2}{\@glo@text}%

```

```

6293     }%
6294 }%

```

Indicate that this entry has now been used

```

6295     \glsunset{#2}%
6296 }%
6297 }

```

\@GLSp1@

```

6298 \def\@GLSp1@#1#2[#3]{%
6299     \glsdoifexists{#2}%
6300     {%
6301         \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

6302     \def\@gls@link@opts{#1}%
6303     \def\@gls@link@label{#2}%

```

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

```

6304     \ifglsused{#2}%
6305     {%
6306         \def\@glo@text{\csname gls@\@glo@type @display\endcsname
6307             {\glspluralaccessdisplay{\glsentryplural{#2}}{#2}}%
6308             {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
6309             {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
6310             {#3}}%
6311     }%
6312     {%
6313         \def\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
6314             {\glsfirstpluralaccessdisplay{\glsentryfirstplural{#2}}{#2}}%
6315             {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
6316             {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
6317             {#3}}%
6318     }%

```

Call \@gls@link If footnote package option has been used, suppress hyperlink for first use.

```

6319     \ifglsused{#2}%
6320     {%
6321         \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
6322     }%
6323     {%
6324         \gls@checkisacronymlist\@glo@type
6325         \ifthenelse{(\boolean{glsisacronymlist}\AND
6326             \boolean{glsacrfootnote})\OR\not\boolean{glshyperfirst}}%
6327         {%
6328             \@gls@link[#1,hyper=false]{#2}{\MakeUppercase{\@glo@text}}%
6329         }%
6330     }%
6331     \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
6332 }%

```

```

6333 }%
        Indicate that this entry has now been used
6334 \glsunset{#2}%
6335 }%
6336 }

```

\@acrshort

```

6337 \def\@acrshort#1#2[#3]{%
6338   \glsdoifexists{#2}%
6339   {%
6340     \edef\@glo@type{\glsentrytype{#2}}%
        Determine what the link text should be (this is stored in \@glo@text)
6341     \def\@glo@text{%
6342       \glsshortaccessdisplay{\glsentryshort{#2}}{#2}%
6343     }%
        Call \@gls@link
6344     \@gls@link[#1]{#2}{\acronymfont{\@glo@text}#3}%
6345   }%
6346 }

```

\@Acrshort

```

6347 \def\@Acrshort#1#2[#3]{%
6348   \glsdoifexists{#2}%
6349   {%
6350     \edef\@glo@type{\glsentrytype{#2}}%
        Determine what the link text should be (this is stored in \@glo@text)
6351     \def\@glo@text{%
6352       \glsshortaccessdisplay{\Glsentryshort{#2}}{#2}%
6353     }%
        Call \@gls@link
6354     \@gls@link[#1]{#2}{\acronymfont{\@glo@text}#3}%
6355   }%
6356 }

```

\@ACRshort

```

6357 \def\@ACRshort#1#2[#3]{%
6358   \glsdoifexists{#2}%
6359   {%
6360     \edef\@glo@type{\glsentrytype{#2}}%
        Determine what the link text should be (this is stored in \@glo@text)
6361     \def\@glo@text{%
6362       \glsshortaccessdisplay{\MakeUppercase{\glsentryshort{#2}}}{#2}%
6363     }%

```

```

Call \@gls@link
6364   \@gls@link[#1]{#2}{\acronymfont{\@glo@text#3}}%
6365   }%
6366 }

```

\@acrlong

```

6367 \def\@acrlong#1#2[#3]{%
6368   \glsdoifexists{#2}%
6369   {%
6370     \edef\@glo@type{\glsentrytype{#2}}%
        Determine what the link text should be (this is stored in \@glo@text)
6371     \def\@glo@text{%
6372       \glslongaccessdisplay{\glsentrylong{#2}}{#2}%
6373     }%
        Call \@gls@link
6374     \@gls@link[#1]{#2}{\@glo@text#3}%
6375     }%
6376 }

```

\@Acrlong

```

6377 \def\@Acrlong#1#2[#3]{%
6378   \glsdoifexists{#2}%
6379   {%
6380     \edef\@glo@type{\glsentrytype{#2}}%
        Determine what the link text should be (this is stored in \@glo@text)
6381     \def\@glo@text{%
6382       \glslongaccessdisplay{\Glsentrylong{#2}}{#2}%
6383     }%
        Call \@gls@link
6384     \@gls@link[#1]{#2}{\@glo@text#3}%
6385     }%
6386 }

```

\@ACRlong

```

6387 \def\@ACRlong#1#2[#3]{%
6388   \glsdoifexists{#2}%
6389   {%
6390     \edef\@glo@type{\glsentrytype{#2}}%
        Determine what the link text should be (this is stored in \@glo@text)
6391     \def\@glo@text{%
6392       \glslongaccessdisplay{\MakeUppercase{\glsentrylong{#2}}}{#2}%
6393     }%
        Call \@gls@link
6394     \@gls@link[#1]{#2}{\@glo@text#3}%
6395     }%
6396 }

```

5.3 Displaying the Glossary

Entries within the glossary or list of acronyms are now formatted via `\accsuppglossaryentryfield` and `\accsuppglossarysubentryfield`.

@glossaryentryfield

```
6397 \ifglxindy
6398   \renewcommand*{\@glossaryentryfield}{%
6399     \string\accsuppglossaryentryfield}
6400 \else
6401   \renewcommand*{\@glossaryentryfield}{%
6402     \string\accsuppglossaryentryfield}
6403 \fi
```

ossarysubentryfield

```
6404 \ifglxindy
6405   \renewcommand*{\@glossarysubentryfield}{%
6406     \string\accsuppglossarysubentryfield}
6407 \else
6408   \renewcommand*{\@glossarysubentryfield}{%
6409     \string\accsuppglossarysubentryfield}
6410 \fi
```

pglossaryentryfield

```
6411 \newcommand*{\accsuppglossaryentryfield}[5]{%
6412   \glossaryentryfield{#1}%
6413   {\glxnameaccessdisplay{#2}{#1}}%
6414   {\glxdescriptionaccessdisplay{#3}{#1}}%
6415   {\glxsymbolaccessdisplay{#4}{#1}}{#5}%
6416 }
```

ossarysubentryfield

```
6417 \newcommand*{\accsuppglossarysubentryfield}[6]{%
6418   \glossaryentryfield{#1}{#2}%
6419   {\glxnameaccessdisplay{#3}{#2}}%
6420   {\glxdescriptionaccessdisplay{#4}{#2}}%
6421   {\glxsymbolaccessdisplay{#5}{#2}}{#6}%
6422 }
```

5.4 Acronyms

Use `\newacronymhook` to modify the key list to set the access text to the long version by default.

```
6423 \renewcommand*{\newacronymhook}{%
6424   \edef\@gls@keylist{shortaccess=\the\glxlongtok,%
6425     \the\glxkeylisttok}%
6426   \expandafter\glxkeylisttok\expandafter{\@gls@keylist}%
6427 }
```

efaultNewAcronymDef Modify default style to use access text:

```

6428 \renewcommand*{\DefaultNewAcronymDef}{%
6429   \edef\@do@newglossaryentry{%
6430     \noexpand\newglossaryentry{\the\glslabeltok}%
6431     {%
6432       type=\acronymtype,%
6433       name={\the\glsshorttok},%
6434       description={\the\glslongtok},%
6435       descriptionaccess=\relax,
6436       text={\the\glsshorttok},%
6437       access={\noexpand\@glo@textaccess},%
6438       sort={\the\glsshorttok},%
6439       short={\the\glsshorttok},%
6440       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6441       shortaccess={\the\glslongtok},%
6442       long={\the\glslongtok},%
6443       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6444       descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6445       first={\noexpand\glslongaccessdisplay
6446         {\the\glslongtok}{\the\glslabeltok}\space
6447         (\noexpand\glsshortaccessdisplay
6448         {\the\glsshorttok}{\the\glslabeltok})},%
6449       plural={\the\glsshorttok\acrpluralsuffix},%
6450       firstplural={\noexpand\glslongpluralaccessdisplay
6451         {\noexpand\@glo@longpl}{\the\glslabeltok}\space
6452         (\noexpand\glsshortpluralaccessdisplay
6453         {\noexpand\@glo@shortpl}{\the\glslabeltok})},%
6454       firstaccess=\relax,
6455       firstpluralaccess=\relax,
6456       textaccess={\noexpand\@glo@shortaccess},%
6457       \the\glskeylisttok
6458     }%
6459   }%
6460   \@do@newglossaryentry
6461 }

```

otnoteNewAcronymDef

```

6462 \renewcommand*{\DescriptionFootnoteNewAcronymDef}{%
6463   \edef\@do@newglossaryentry{%
6464     \noexpand\newglossaryentry{\the\glslabeltok}%
6465     {%
6466       type=\acronymtype,%
6467       name={\noexpand\acronymfont{\the\glsshorttok}},%
6468       sort={\the\glsshorttok},%
6469       text={\the\glsshorttok},%
6470       short={\the\glsshorttok},%
6471       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6472       shortaccess={\the\glslongtok},%
6473       long={\the\glslongtok},%

```



```

6474     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6475     access={\noexpand\@glo@textaccess},%
6476     plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6477     symbol={\the\glslongtok},%
6478     symbolplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6479     firstpluralaccess=\relax,
6480     textaccess={\noexpand\@glo@shortaccess},%
6481     \the\glskeylisttok
6482 }%
6483 }%
6484 \@do@newglossaryentry
6485 }

```

ptionNewAcronymDef

```

6486 \renewcommand*{\DescriptionNewAcronymDef}{%
6487   \edef\@do@newglossaryentry{%
6488     \noexpand\newglossaryentry{\the\glslabeltok}%
6489     {%
6490       type=\acronymtype,%
6491       name={\noexpand
6492         \acrnameformat{\the\glsshorttok}{\the\glslongtok}},%
6493       access={\noexpand\@glo@textaccess},%
6494       sort={\the\glsshorttok},%
6495       short={\the\glsshorttok},%
6496       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6497       shortaccess={\the\glslongtok},%
6498       long={\the\glslongtok},%
6499       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6500       first={\the\glslongtok},%
6501       firstaccess=\relax,
6502       firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6503       text={\the\glsshorttok},%
6504       textaccess={\the\glslongtok},%
6505       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6506       symbol={\noexpand\@glo@text},%
6507       symbolaccess={\noexpand\@glo@textaccess},%
6508       symbolplural={\noexpand\@glo@plural},%
6509       firstpluralaccess=\relax,
6510       textaccess={\noexpand\@glo@shortaccess},%
6511       \the\glskeylisttok}%
6512   }%
6513   \@do@newglossaryentry
6514 }

```

otnoteNewAcronymDef

```

6515 \renewcommand*{\FootnoteNewAcronymDef}{%
6516   \edef\@do@newglossaryentry{%
6517     \noexpand\newglossaryentry{\the\glslabeltok}%
6518     {%

```

```

6519     type=\acronymtype,%
6520     name={\noexpand\acronymfont{\the\glsshorttok}},%
6521     sort={\the\glsshorttok},%
6522     text={\the\glsshorttok},%
6523     textaccess={\the\glslongtok},%
6524     access={\noexpand\@glo@textaccess},%
6525     plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6526     short={\the\glsshorttok},%
6527     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6528     long={\the\glslongtok},%
6529     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6530     description={\the\glslongtok},%
6531     descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6532     \the\glskeylisttok
6533 }%
6534 }%
6535 \@do@newglossaryentry
6536 }

```

\SmallNewAcronymDef

```

6537 \renewcommand*{\SmallNewAcronymDef}{%
6538   \edef\@do@newglossaryentry{%
6539     \noexpand\newglossaryentry{\the\glslabeltok}%
6540     {%
6541       type=\acronymtype,%
6542       name={\noexpand\acronymfont{\the\glsshorttok}},%
6543       access={\noexpand\@glo@symbolaccess},%
6544       sort={\the\glsshorttok},%
6545       short={\the\glsshorttok},%
6546       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6547       shortaccess={\the\glslongtok},%
6548       long={\the\glslongtok},%
6549       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6550       text={\noexpand\@glo@short},%
6551       textaccess={\noexpand\@glo@shortaccess},%
6552       plural={\noexpand\@glo@shortpl},%
6553       first={\the\glslongtok},%
6554       firstaccess=\relax,
6555       firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6556       description={\noexpand\@glo@first},%
6557       descriptionplural={\noexpand\@glo@firstplural},%
6558       symbol={\the\glsshorttok},%
6559       symbolaccess={\the\glslongtok},%
6560       symbolplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6561       \the\glskeylisttok
6562     }%
6563   }%
6564   \@do@newglossaryentry
6565 }

```

The following are kept for compatibility with versions before 3.0:

```
\glsshortaccesskey
6566 \newcommand*{\glsshortaccesskey}{\glsshortkey access}%

shortpluralaccesskey
6567 \newcommand*{\glsshortpluralaccesskey}{\glsshortpluralkey access}%

\glslongaccesskey
6568 \newcommand*{\glslongaccesskey}{\glslongkey access}%

longpluralaccesskey
6569 \newcommand*{\glslongpluralaccesskey}{\glslongpluralkey access}%
```

5.5 Debugging Commands

```
\showglonameaccess
6570 \newcommand*{\showglonameaccess}[1]{%
6571 \expandafter\show\csname glo@#1@textaccess\endcsname
6572 }

\showglotextaccess
6573 \newcommand*{\showglotextaccess}[1]{%
6574 \expandafter\show\csname glo@#1@textaccess\endcsname
6575 }

showglopluralaccess
6576 \newcommand*{\showglopluralaccess}[1]{%
6577 \expandafter\show\csname glo@#1@pluralaccess\endcsname
6578 }

\showglofirstaccess
6579 \newcommand*{\showglofirstaccess}[1]{%
6580 \expandafter\show\csname glo@#1@firstaccess\endcsname
6581 }

lofirstpluralaccess
6582 \newcommand*{\showglofirstpluralaccess}[1]{%
6583 \expandafter\show\csname glo@#1@firstpluralaccess\endcsname
6584 }

showglosymbolaccess
6585 \newcommand*{\showglosymbolaccess}[1]{%
6586 \expandafter\show\csname glo@#1@symbolaccess\endcsname
6587 }
```

osymbolpluralaccess

```
6588 \newcommand*{\showglosymbolpluralaccess}[1]{%
6589   \expandafter\show\csname glo@#1@symbolpluralaccess\endcsname
6590 }
```

\showglodescaccess

```
6591 \newcommand*{\showglodescaccess}[1]{%
6592   \expandafter\show\csname glo@#1@descaccess\endcsname
6593 }
```

glodescpluralaccess

```
6594 \newcommand*{\showglodescpluralaccess}[1]{%
6595   \expandafter\show\csname glo@#1@descpluralaccess\endcsname
6596 }
```

\showgloshortaccess

```
6597 \newcommand*{\showgloshortaccess}[1]{%
6598   \expandafter\show\csname glo@#1@shortaccess\endcsname
6599 }
```

loshortpluralaccess

```
6600 \newcommand*{\showgloshortpluralaccess}[1]{%
6601   \expandafter\show\csname glo@#1@shortpluralaccess\endcsname
6602 }
```

\showglolongaccess

```
6603 \newcommand*{\showglolongaccess}[1]{%
6604   \expandafter\show\csname glo@#1@longaccess\endcsname
6605 }
```

glolongpluralaccess

```
6606 \newcommand*{\showglolongpluralaccess}[1]{%
6607   \expandafter\show\csname glo@#1@longpluralaccess\endcsname
6608 }
```

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.

```
6609 \NeedsTeXFormat{LaTeX2e}
6610 \ProvidesPackage{glossaries-babel}[2009/04/16 v1.2 (NLCT)]
```

English:

```
6611 \@ifundefined{captionsenglish}{%}%
6612   \addto\captionsenglish{%
6613     \renewcommand*{\glossaryname}{Glossary}%
6614     \renewcommand*{\acronymname}{Acronyms}%
6615     \renewcommand*{\entryname}{Notation}%
6616     \renewcommand*{\descriptionname}{Description}%
6617     \renewcommand*{\symbolname}{Symbol}%
6618     \renewcommand*{\pagelistname}{Page List}%
6619     \renewcommand*{\glssymbolsgroupname}{Symbols}%
6620     \renewcommand*{\glsnumbersgroupname}{Numbers}%
6621 }%
6622 }
6623 \@ifundefined{captionsamerican}{%}%
6624   \addto\captionsamerican{%
6625     \renewcommand*{\glossaryname}{Glossary}%
6626     \renewcommand*{\acronymname}{Acronyms}%
6627     \renewcommand*{\entryname}{Notation}%
6628     \renewcommand*{\descriptionname}{Description}%
6629     \renewcommand*{\symbolname}{Symbol}%
6630     \renewcommand*{\pagelistname}{Page List}%
6631     \renewcommand*{\glssymbolsgroupname}{Symbols}%
6632     \renewcommand*{\glsnumbersgroupname}{Numbers}%
6633 }%
6634 }
6635 \@ifundefined{captionsaustralian}{%}%
6636   \addto\captionsaustralian{%
6637     \renewcommand*{\glossaryname}{Glossary}%
6638     \renewcommand*{\acronymname}{Acronyms}%
6639     \renewcommand*{\entryname}{Notation}%
6640     \renewcommand*{\descriptionname}{Description}%
6641     \renewcommand*{\symbolname}{Symbol}%
6642     \renewcommand*{\pagelistname}{Page List}%
6643     \renewcommand*{\glssymbolsgroupname}{Symbols}%
6644     \renewcommand*{\glsnumbersgroupname}{Numbers}%
6645 }%
6646 }
6647 \@ifundefined{captionsbritish}{%}%
6648   \addto\captionsbritish{%
6649     \renewcommand*{\glossaryname}{Glossary}%
6650     \renewcommand*{\acronymname}{Acronyms}%
6651     \renewcommand*{\entryname}{Notation}%
6652     \renewcommand*{\descriptionname}{Description}%
6653     \renewcommand*{\symbolname}{Symbol}%
6654     \renewcommand*{\pagelistname}{Page List}%
6655     \renewcommand*{\glssymbolsgroupname}{Symbols}%
6656     \renewcommand*{\glsnumbersgroupname}{Numbers}%
6657 }%
6658 \@ifundefined{captionscanadian}{%}%
```

```

6659 \addto\captionscanadian{%
6660 \renewcommand*{\glossaryname}{Glossary}%
6661 \renewcommand*{\acronymname}{Acronyms}%
6662 \renewcommand*{\entryname}{Notation}%
6663 \renewcommand*{\descriptionname}{Description}%
6664 \renewcommand*{\symbolname}{Symbol}%
6665 \renewcommand*{\pagelistname}{Page List}%
6666 \renewcommand*{\glssymbolsgroupname}{Symbols}%
6667 \renewcommand*{\glsnumbersgroupname}{Numbers}%
6668 }%
6669 }
6670 \@ifundefined{captionsnewzealand}{}{%
6671 \addto\captionscanadian{%
6672 \renewcommand*{\glossaryname}{Glossary}%
6673 \renewcommand*{\acronymname}{Acronyms}%
6674 \renewcommand*{\entryname}{Notation}%
6675 \renewcommand*{\descriptionname}{Description}%
6676 \renewcommand*{\symbolname}{Symbol}%
6677 \renewcommand*{\pagelistname}{Page List}%
6678 \renewcommand*{\glssymbolsgroupname}{Symbols}%
6679 \renewcommand*{\glsnumbersgroupname}{Numbers}%
6680 }%
6681 }
6682 \@ifundefined{captionsUKenglish}{}{%
6683 \addto\captionscanadian{%
6684 \renewcommand*{\glossaryname}{Glossary}%
6685 \renewcommand*{\acronymname}{Acronyms}%
6686 \renewcommand*{\entryname}{Notation}%
6687 \renewcommand*{\descriptionname}{Description}%
6688 \renewcommand*{\symbolname}{Symbol}%
6689 \renewcommand*{\pagelistname}{Page List}%
6690 \renewcommand*{\glssymbolsgroupname}{Symbols}%
6691 \renewcommand*{\glsnumbersgroupname}{Numbers}%
6692 }%
6693 }
6694 \@ifundefined{captionsUSenglish}{}{%
6695 \addto\captionscanadian{%
6696 \renewcommand*{\glossaryname}{Glossary}%
6697 \renewcommand*{\acronymname}{Acronyms}%
6698 \renewcommand*{\entryname}{Notation}%
6699 \renewcommand*{\descriptionname}{Description}%
6700 \renewcommand*{\symbolname}{Symbol}%
6701 \renewcommand*{\pagelistname}{Page List}%
6702 \renewcommand*{\glssymbolsgroupname}{Symbols}%
6703 \renewcommand*{\glsnumbersgroupname}{Numbers}%
6704 }%
6705 }

```

German (quite a few variations were suggested for German; I settled on the following):

```

6706 \@ifundefined{captionsgerman}{%
6707   \addto\captionsgerman{%
6708     \renewcommand*{\glossaryname}{Glossar}%
6709     \renewcommand*{\acronymname}{Akronyme}%
6710     \renewcommand*{\entryname}{Bezeichnung}%
6711     \renewcommand*{\descriptionname}{Beschreibung}%
6712     \renewcommand*{\symbolname}{Symbol}%
6713     \renewcommand*{\pagelistname}{Seiten}%
6714     \renewcommand*{\glssymbolsgroupname}{Symbole}%
6715     \renewcommand*{\glsnumbersgroupname}{Zahlen}}
6716 }
```

ngerman is identical to German:

```

6717 \@ifundefined{captionsgerman}{%
6718   \addto\captionsgerman{%
6719     \renewcommand*{\glossaryname}{Glossar}%
6720     \renewcommand*{\acronymname}{Akronyme}%
6721     \renewcommand*{\entryname}{Bezeichnung}%
6722     \renewcommand*{\descriptionname}{Beschreibung}%
6723     \renewcommand*{\symbolname}{Symbol}%
6724     \renewcommand*{\pagelistname}{Seiten}%
6725     \renewcommand*{\glssymbolsgroupname}{Symbole}%
6726     \renewcommand*{\glsnumbersgroupname}{Zahlen}}
6727 }
```

Italian:

```

6728 \@ifundefined{captionssitalian}{%
6729   \addto\captionssitalian{%
6730     \renewcommand*{\glossaryname}{Glossario}%
6731     \renewcommand*{\acronymname}{Acronimi}%
6732     \renewcommand*{\entryname}{Nomenclatura}%
6733     \renewcommand*{\descriptionname}{Descrizione}%
6734     \renewcommand*{\symbolname}{Simbolo}%
6735     \renewcommand*{\pagelistname}{Elenco delle pagine}%
6736     \renewcommand*{\glssymbolsgroupname}{Simboli}%
6737     \renewcommand*{\glsnumbersgroupname}{Numeri}}
6738 }
```

Dutch:

```

6739 \@ifundefined{captionsdutch}{%
6740   \addto\captionsdutch{%
6741     \renewcommand*{\glossaryname}{Woordenlijst}%
6742     \renewcommand*{\acronymname}{Acroniemen}%
6743     \renewcommand*{\entryname}{Benaming}%
6744     \renewcommand*{\descriptionname}{Beschrijving}%
6745     \renewcommand*{\symbolname}{Symbool}%
6746     \renewcommand*{\pagelistname}{Pagina's}%
6747     \renewcommand*{\glssymbolsgroupname}{Symbolen}%

```

```

6748 \renewcommand*{\glsnumbersgroupname}{Cijfers}}
6749 }

Spanish:

6750 \@ifundefined{captionsspanish}{}{%
6751 \addto\captionsspanish{%
6752 \renewcommand*{\glossaryname}{Glosario}%
6753 \renewcommand*{\acronymname}{Siglas}%
6754 \renewcommand*{\entryname}{Entrada}%
6755 \renewcommand*{\descriptionname}{Descripci\`on}%
6756 \renewcommand*{\symbolname}{S\`{i}mbolo}%
6757 \renewcommand*{\pagelistname}{Lista de p\`aginas}%
6758 \renewcommand*{\glssymbolsgroupname}{S\`{i}mbolos}%
6759 \renewcommand*{\glsnumbersgroupname}{N\`umeros}}
6760 }

French:

6761 \@ifundefined{captionsfrench}{}{%
6762 \addto\captionsfrench{%
6763 \renewcommand*{\glossaryname}{Glossaire}%
6764 \renewcommand*{\acronymname}{Acronymes}%
6765 \renewcommand*{\entryname}{Terme}%
6766 \renewcommand*{\descriptionname}{Description}%
6767 \renewcommand*{\symbolname}{Symbole}%
6768 \renewcommand*{\pagelistname}{Pages}%
6769 \renewcommand*{\glssymbolsgroupname}{Symboles}%
6770 \renewcommand*{\glsnumbersgroupname}{Nombres}}
6771 }

6772 \@ifundefined{captionsfrenchb}{}{%
6773 \addto\captionsfrenchb{%
6774 \renewcommand*{\glossaryname}{Glossaire}%
6775 \renewcommand*{\acronymname}{Acronymes}%
6776 \renewcommand*{\entryname}{Terme}%
6777 \renewcommand*{\descriptionname}{Description}%
6778 \renewcommand*{\symbolname}{Symbole}%
6779 \renewcommand*{\pagelistname}{Pages}%
6780 \renewcommand*{\glssymbolsgroupname}{Symboles}%
6781 \renewcommand*{\glsnumbersgroupname}{Nombres}}
6782 }

6783 \@ifundefined{captionsfrancais}{}{%
6784 \addto\captionsfrancais{%
6785 \renewcommand*{\glossaryname}{Glossaire}%
6786 \renewcommand*{\acronymname}{Acronymes}%
6787 \renewcommand*{\entryname}{Terme}%
6788 \renewcommand*{\descriptionname}{Description}%
6789 \renewcommand*{\symbolname}{Symbole}%
6790 \renewcommand*{\pagelistname}{Pages}%
6791 \renewcommand*{\glssymbolsgroupname}{Symboles}%
6792 \renewcommand*{\glsnumbersgroupname}{Nombres}}
6793 }

```


Danish:

```
6794 \@ifundefined{captionsdanish}{%}{%
6795   \addto\captionsdanish{%
6796     \renewcommand*{\glossaryname}{Ordliste}%
6797     \renewcommand*{\acronymname}{Akronymer}%
6798     \renewcommand*{\entryname}{Symbolforklaring}%
6799     \renewcommand*{\descriptionname}{Beskrivelse}%
6800     \renewcommand*{\symbolname}{Symbol}%
6801     \renewcommand*{\pagelistname}{Side}%
6802     \renewcommand*{\glssymbolsgroupname}{Symboler}%
6803     \renewcommand*{\glsnumbersgroupname}{Tal}}
6804 }
```

Irish:

```
6805 \@ifundefined{captionsirish}{%}{%
6806   \addto\captionsirish{%
6807     \renewcommand*{\glossaryname}{Gluais}%
6808     \renewcommand*{\acronymname}{Acrainmneacha}%
```

wasn't sure whether to go for Nóta (Note), Ciall ('Meaning', 'sense') or Brí ('Meaning'). In the end I chose Ciall.

```
6809     \renewcommand*{\entryname}{Ciall}%
6810     \renewcommand*{\descriptionname}{Tuairisc}%
```

Again, not sure whether to use Comhartha/Comharthaí or Siombail/Siombaile, so have chosen the former.

```
6811     \renewcommand*{\symbolname}{Comhartha}%
6812     \renewcommand*{\glssymbolsgroupname}{Comhartha\'}{\i}}%
6813     \renewcommand*{\pagelistname}{Leathanaigh}%
6814     \renewcommand*{\glsnumbersgroupname}{Uimhreacha}}
6815 }
```

Hungarian:

```
6816 \@ifundefined{captionsmagyar}{%}{%
6817   \addto\captionsmagyar{%
6818     \renewcommand*{\glossaryname}{Sz\'ojegyz\'ek}%
6819     \renewcommand*{\acronymname}{Bet\H uszavak}%
6820     \renewcommand*{\entryname}{Kifejez\'es}%
6821     \renewcommand*{\descriptionname}{Magyar\'azat}%
6822     \renewcommand*{\symbolname}{Jel\'ol\'es}%
6823     \renewcommand*{\pagelistname}{Oldalsz\'am}%
6824     \renewcommand*{\glssymbolsgroupname}{Jelek}%
6825     \renewcommand*{\glsnumbersgroupname}{Sz\'amjegyek}%
6826   }
6827 }
6828 \@ifundefined{captionshungarian}{%}{%
6829   \addto\captionshungarian{%
6830     \renewcommand*{\glossaryname}{Sz\'ojegyz\'ek}%
6831     \renewcommand*{\acronymname}{Bet\H uszavak}%
6832     \renewcommand*{\entryname}{Kifejez\'es}%
6833     \renewcommand*{\descriptionname}{Magyar\'azat}%
```

```

6834 \renewcommand*{\symbolname}{Jel\`ol\`es}%
6835 \renewcommand*{\pagelistname}{Oldalsz\`am}%
6836 \renewcommand*{\glssymbolsgroupname}{Jelek}%
6837 \renewcommand*{\glsnumbersgroupname}{Sz\`amjegyek}%
6838 }
6839 }

```

Polish

```

6840 \@ifundefined{captionspolish}{}{%
6841 \addto\captionspolish{%
6842 \renewcommand*{\glossaryname}{S\`lownik termin\`ow}%
6843 \renewcommand*{\acronymname}{Skr\`ot}%
6844 \renewcommand*{\entryname}{Termin}%
6845 \renewcommand*{\descriptionname}{Opis}%
6846 \renewcommand*{\symbolname}{Symbol}%
6847 \renewcommand*{\pagelistname}{Strony}%
6848 \renewcommand*{\glssymbolsgroupname}{Symbole}%
6849 \renewcommand*{\glsnumbersgroupname}{Liczby}}
6850 }

```

Brazilian

```

6851 \@ifundefined{captionsbrazil}{}{%
6852 \addto\captionsbrazil{%
6853 \renewcommand*{\glossaryname}{Gloss\`ario}%
6854 \renewcommand*{\acronymname}{Siglas}%
6855 \renewcommand*{\entryname}{Nota\`c c\`ao}%
6856 \renewcommand*{\descriptionname}{Descri\`c c\`ao}%
6857 \renewcommand*{\symbolname}{S\`imbolo}%
6858 \renewcommand*{\pagelistname}{Lista de P\`aginas}%
6859 \renewcommand*{\glssymbolsgroupname}{S\`imbolos}%
6860 \renewcommand*{\glsnumbersgroupname}{N\`umeros}%
6861 }%
6862 }

```

6.2 Polyglossia Captions

```

6863 \NeedsTeXFormat{LaTeX2e}
6864 \ProvidesPackage{glossaries-polyglossia}[2009/11/09 v1.0 (NLCT)]

```

English:

```

6865 \@ifundefined{captionseenglish}{}{%
6866 \expandafter\toks@\expandafter{\captionseenglish
6867 \renewcommand*{\glossaryname}{\textenglish{Glossary}}%
6868 \renewcommand*{\acronymname}{\textenglish{Acronyms}}%
6869 \renewcommand*{\entryname}{\textenglish{Notation}}%
6870 \renewcommand*{\descriptionname}{\textenglish{Description}}%
6871 \renewcommand*{\symbolname}{\textenglish{Symbol}}%
6872 \renewcommand*{\pagelistname}{\textenglish{Page List}}%
6873 \renewcommand*{\glssymbolsgroupname}{\textenglish{Symbols}}%
6874 \renewcommand*{\glsnumbersgroupname}{\textenglish{Numbers}}%
6875 }%

```

```

6876 \edef\captionseenglish{\the\toks@}%
6877 }

```

German:

```

6878 \@ifundefined{captionsgerman}{}{%
6879 \expandafter\toks@\expandafter{\captionsgerman
6880 \renewcommand*{\glossaryname}{\textgerman{Glossar}}}%
6881 \renewcommand*{\acronymname}{\textgerman{Akronyme}}}%
6882 \renewcommand*{\entryname}{\textgerman{Bezeichnung}}}%
6883 \renewcommand*{\descriptionname}{\textgerman{Beschreibung}}}%
6884 \renewcommand*{\symbolname}{\textgerman{Symbol}}}%
6885 \renewcommand*{\pagelistname}{\textgerman{Seiten}}}%
6886 \renewcommand*{\glssymbolsgroupname}{\textgerman{Symbole}}}%
6887 \renewcommand*{\glsnumbersgroupname}{\textgerman{Zahlen}}}%
6888 }%
6889 \edef\captionsgerman{\the\toks@}%
6890 }

```

Italian:

```

6891 \@ifundefined{captionssitalian}{}{%
6892 \expandafter\toks@\expandafter{\captionssitalian
6893 \renewcommand*{\glossaryname}{\textitalian{Glossario}}}%
6894 \renewcommand*{\acronymname}{\textitalian{Acronimi}}}%
6895 \renewcommand*{\entryname}{\textitalian{Nomenclatura}}}%
6896 \renewcommand*{\descriptionname}{\textitalian{Descrizione}}}%
6897 \renewcommand*{\symbolname}{\textitalian{Simbolo}}}%
6898 \renewcommand*{\pagelistname}{\textitalian{Elenco delle pagine}}}%
6899 \renewcommand*{\glssymbolsgroupname}{\textitalian{Simboli}}}%
6900 \renewcommand*{\glsnumbersgroupname}{\textitalian{Numeri}}}%
6901 }%
6902 \edef\captionssitalian{\the\toks@}%
6903 }

```

Dutch:

```

6904 \@ifundefined{captionsdutch}{}{%
6905 \expandafter\toks@\expandafter{\captionsdutch
6906 \renewcommand*{\glossaryname}{\textdutch{Woordenlijst}}}%
6907 \renewcommand*{\acronymname}{\textdutch{Acroniemen}}}%
6908 \renewcommand*{\entryname}{\textdutch{Benaming}}}%
6909 \renewcommand*{\descriptionname}{\textdutch{Beschrijving}}}%
6910 \renewcommand*{\symbolname}{\textdutch{Symbool}}}%
6911 \renewcommand*{\pagelistname}{\textdutch{Pagina's}}}%
6912 \renewcommand*{\glssymbolsgroupname}{\textdutch{Symbolen}}}%
6913 \renewcommand*{\glsnumbersgroupname}{\textdutch{Cijfers}}}%
6914 }%
6915 \edef\captionsdutch{\the\toks@}%
6916 }

```

Spanish:

```

6917 \@ifundefined{captionssspanish}{}{%
6918 \expandafter\toks@\expandafter{\captionssspanish
6919 \renewcommand*{\glossaryname}{\textspanish{Glosario}}}%

```

```

6920 \renewcommand*{\acronymname}{\textspanish{Siglas}}%
6921 \renewcommand*{\entryname}{\textspanish{Entrada}}%
6922 \renewcommand*{\descriptionname}{\textspanish{Descripci' on}}%
6923 \renewcommand*{\symbolname}{\textspanish{S\'\i mbolo}}%
6924 \renewcommand*{\pagelistname}{\textspanish{Lista de p\ ' aginas}}%
6925 \renewcommand*{\glssymbolsgroupname}{\textspanish{S\'\i mbolos}}%
6926 \renewcommand*{\glslnumbersgroupname}{\textspanish{N\ ' umeros}}%
6927 }%
6928 \edef\captionsspanish{\the\toks@}%
6929 }

```

French:

```

6930 \@ifundefined{captionsfrench}{}{%
6931 \expandafter\toks@\expandafter{\captionsfrench
6932 \renewcommand*{\glossaryname}{\textfrench{Glossaire}}%
6933 \renewcommand*{\acronymname}{\textfrench{Acronymes}}%
6934 \renewcommand*{\entryname}{\textfrench{Terme}}%
6935 \renewcommand*{\descriptionname}{\textfrench{Description}}%
6936 \renewcommand*{\symbolname}{\textfrench{Symbole}}%
6937 \renewcommand*{\pagelistname}{\textfrench{Pages}}%
6938 \renewcommand*{\glssymbolsgroupname}{\textfrench{Symboles}}%
6939 \renewcommand*{\glslnumbersgroupname}{\textfrench{Nombres}}%
6940 }%
6941 \edef\captionsfrench{\the\toks@}%
6942 }

```

Danish:

```

6943 \@ifundefined{captionsdanish}{}{%
6944 \expandafter\toks@\expandafter{\captionsdanish
6945 \renewcommand*{\glossaryname}{\textdanish{Ordliste}}%
6946 \renewcommand*{\acronymname}{\textdanish{Akronymer}}%
6947 \renewcommand*{\entryname}{\textdanish{Symbolforklaring}}%
6948 \renewcommand*{\descriptionname}{\textdanish{Beskrivelse}}%
6949 \renewcommand*{\symbolname}{\textdanish{Symbol}}%
6950 \renewcommand*{\pagelistname}{\textdanish{Side}}%
6951 \renewcommand*{\glssymbolsgroupname}{\textdanish{Symboler}}%
6952 \renewcommand*{\glslnumbersgroupname}{\textdanish{Tal}}%
6953 }%
6954 \edef\captionsdanish{\the\toks@}%
6955 }

```

Irish:

```

6956 \@ifundefined{captionsirish}{}{%
6957 \expandafter\toks@\expandafter{\captionsirish
6958 \renewcommand*{\glossaryname}{\textirish{Gluais}}%
6959 \renewcommand*{\acronymname}{\textirish{Acrainmneacha}}%
6960 \renewcommand*{\entryname}{\textirish{Ciall}}%
6961 \renewcommand*{\descriptionname}{\textirish{Tuaireasc}}%
6962 \renewcommand*{\symbolname}{\textirish{Comhartha}}%
6963 \renewcommand*{\glssymbolsgroupname}{\textirish{Comhartha\ ' \i}}%
6964 \renewcommand*{\pagelistname}{\textirish{Leathanaigh}}%

```

```

6965 \renewcommand*{\glnumbersgroupname}{\textirish{Uimhreacha}}}%
6966 }%
6967 \edef\captionisirish{\the\toks@}%
6968 }

```

Hungarian:

```

6969 \@ifundefined{captionsmagyar}{}{%
6970 \expandafter\toks@\expandafter{\captionsmagyar
6971 \renewcommand*{\glossaryname}{\textmagyar{Sz\'ojegyz\'ek}}}%
6972 \renewcommand*{\acronymname}{\textmagyar{Bet\H uszavak}}}%
6973 \renewcommand*{\entryname}{\textmagyar{Kifejez\'es}}}%
6974 \renewcommand*{\descriptionname}{\textmagyar{Magyar\'azat}}}%
6975 \renewcommand*{\symbolname}{\textmagyar{Jel\'ol\'es}}}%
6976 \renewcommand*{\pagelistname}{\textmagyar{Oldalsz\'am}}}%
6977 \renewcommand*{\glssymbolsgroupname}{\textmagyar{Jelek}}}%
6978 \renewcommand*{\glnumbersgroupname}{\textmagyar{Sz\'amjegyek}}}%
6979 }%
6980 \edef\captionsmagyar{\the\toks@}%
6981 }

```

Polish

```

6982 \@ifundefined{captionspolish}{}{%
6983 \expandafter\toks@\expandafter{\captionspolish
6984 \renewcommand*{\glossaryname}{\textpolish{S{\l}ownik termin\'ow}}}%
6985 \renewcommand*{\acronymname}{\textpolish{Skr\'ot}}}%
6986 \renewcommand*{\entryname}{\textpolish{Termin}}}%
6987 \renewcommand*{\descriptionname}{\textpolish{Opis}}}%
6988 \renewcommand*{\symbolname}{\textpolish{Symbol}}}%
6989 \renewcommand*{\pagelistname}{\textpolish{Strony}}}%
6990 \renewcommand*{\glssymbolsgroupname}{\textpolish{Symbole}}}%
6991 \renewcommand*{\glnumbersgroupname}{\textpolish{Liczby}}}%
6992 }%
6993 \edef\captionspolish{\the\toks@}%
6994 }

```

Portugues

```

6995 \@ifundefined{captionsportuges}{}{%
6996 \expandafter\toks@\expandafter{\captionsportuges
6997 \renewcommand*{\glossaryname}{\textportuges{Gloss\'ario}}}%
6998 \renewcommand*{\acronymname}{\textportuges{Siglas}}}%
6999 \renewcommand*{\entryname}{\textportuges{Nota\c c\~ao}}}%
7000 \renewcommand*{\descriptionname}{\textportuges{Descri\c c\~ao}}}%
7001 \renewcommand*{\symbolname}{\textportuges{S\'imbolo}}}%
7002 \renewcommand*{\pagelistname}{\textportuges{Lista de P\'aginas}}}%
7003 \renewcommand*{\glssymbolsgroupname}{\textportuges{S\'imbolos}}}%
7004 \renewcommand*{\glnumbersgroupname}{\textportuges{N\'umeros}}}%
7005 }%
7006 \edef\captionsportuges{\the\toks@}%
7007 }

```

6.3 Brazilian Dictionary

This is a dictionary file provided by Thiago de Melo for use with the package.

```
7008 \ProvidesDictionary{glossaries-dictionary}{Brazilian}
```

Provide Brazilian translations:

```
7009 \providetranslation{Glossary}{Gloss\`ario}
7010 \providetranslation{Acronyms}{Siglas}
7011 \providetranslation{Notation (glossaries)}{Nota\c c\~ao}
7012 \providetranslation{Description (glossaries)}{Descri\c c\~ao}
7013 \providetranslation{Symbol (glossaries)}{S\`imbolo}
7014 \providetranslation{Page List (glossaries)}{Lista de P\`aginas}
7015 \providetranslation{Symbols (glossaries)}{S\`imbolos}
7016 \providetranslation{Numbers (glossaries)}{N\`umeros}
```

6.4 Danish Dictionary

This is a dictionary file provided for use with the package.

```
7017 \ProvidesDictionary{glossaries-dictionary}{Danish}
```

Provide Danish translations:

```
7018 \providetranslation{Glossary}{Ordliste}
7019 \providetranslation{Acronyms}{Akronymer}
7020 \providetranslation{Notation (glossaries)}{Symbolforklaring}
7021 \providetranslation{Description (glossaries)}{Beskrivelse}
7022 \providetranslation{Symbol (glossaries)}{Symbol}
7023 \providetranslation{Page List (glossaries)}{Side}
7024 \providetranslation{Symbols (glossaries)}{Symboler}
7025 \providetranslation{Numbers (glossaries)}{Tal}
```

6.5 Dutch Dictionary

This is a dictionary file provided for use with the package.

```
7026 \ProvidesDictionary{glossaries-dictionary}{Dutch}
```

Provide Dutch translations:

```
7027 \providetranslation{Glossary}{Woordenlijst}
7028 \providetranslation{Acronyms}{Acroniemen}
7029 \providetranslation{Notation (glossaries)}{Benaming}
7030 \providetranslation{Description (glossaries)}{Beschrijving}
7031 \providetranslation{Symbol (glossaries)}{Symbool}
7032 \providetranslation{Page List (glossaries)}{Pagina's}
7033 \providetranslation{Symbols (glossaries)}{Symbolen}
7034 \providetranslation{Numbers (glossaries)}{Cijfers}
```

6.6 English Dictionary

This is a dictionary file provided for use with the package.

```
7035 \ProvidesDictionary{glossaries-dictionary}{English}
```

Provide English translations:

```
7036 \providetranslation{Glossary}{Glossary}
7037 \providetranslation{Acronyms}{Acronyms}
7038 \providetranslation{Notation (glossaries)}{Notation}
7039 \providetranslation{Description (glossaries)}{Description}
7040 \providetranslation{Symbol (glossaries)}{Symbol}
7041 \providetranslation{Page List (glossaries)}{Page List}
7042 \providetranslation{Symbols (glossaries)}{Symbols}
7043 \providetranslation{Numbers (glossaries)}{Numbers}
```

6.7 French Dictionary

This is a dictionary file provided for use with the package.

```
7044 \ProvidesDictionary{glossaries-dictionary}{French}
```

Provide French translations:

```
7045 \providetranslation{Glossary}{Glossaire}
7046 \providetranslation{Acronyms}{Acronymes}
7047 \providetranslation{Notation (glossaries)}{Terme}
7048 \providetranslation{Description (glossaries)}{Description}
7049 \providetranslation{Symbol (glossaries)}{Symbole}
7050 \providetranslation{Page List (glossaries)}{Pages}
7051 \providetranslation{Symbols (glossaries)}{Symboles}
7052 \providetranslation{Numbers (glossaries)}{Nombres}
```

6.8 German Dictionary

This is a dictionary file provided for use with the package.

```
7053 \ProvidesDictionary{glossaries-dictionary}{German}
```

Provide German translations (quite a few variations were suggested for German; I settled on the following):

```
7054 \providetranslation{Glossary}{Glossar}
7055 \providetranslation{Acronyms}{Akronyme}
7056 \providetranslation{Notation (glossaries)}{Bezeichnung}
7057 \providetranslation{Description (glossaries)}{Beschreibung}
7058 \providetranslation{Symbol (glossaries)}{Symbol}
7059 \providetranslation{Page List (glossaries)}{Seiten}
7060 \providetranslation{Symbols (glossaries)}{Symbole}
7061 \providetranslation{Numbers (glossaries)}{Zahlen}
```

6.9 Irish Dictionary

This is a dictionary file provided for use with the package.

```
7062 \ProvidesDictionary{glossaries-dictionary}{Irish}
```

Provide Irish translations:

```
7063 \providetranslation{Glossary}{Gluais}
7064 \providetranslation{Acronyms}{Acrainmneacha}
```

```

7065 \providetranslation{Notation (glossaries)}{Ciall}
7066 \providetranslation{Description (glossaries)}{Tuairisc}
7067 \providetranslation{Symbol (glossaries)}{Comhartha}
7068 \providetranslation{Page List (glossaries)}{Leathanaigh}
7069 \providetranslation{Symbols (glossaries)}{Comhartha'\{i}}
7070 \providetranslation{Numbers (glossaries)}{Uimhreacha}

```

6.10 Italian Dictionary

This is a dictionary file provided for use with the package.

```

7071 \ProvidesDictionary{glossaries-dictionary}{Italian}

```

Provide Italian translations:

```

7072 \providetranslation{Glossary}{Glossario}
7073 \providetranslation{Acronyms}{Acronimi}
7074 \providetranslation{Notation (glossaries)}{Nomenclatura}
7075 \providetranslation{Description (glossaries)}{Descrizione}
7076 \providetranslation{Symbol (glossaries)}{Simbolo}
7077 \providetranslation{Page List (glossaries)}{Elenco delle pagine}
7078 \providetranslation{Symbols (glossaries)}{Simboli}
7079 \providetranslation{Numbers (glossaries)}{Numeri}

```

6.11 Magyar Dictionary

This is a dictionary file provided for use with the package.

```

7080 \ProvidesDictionary{glossaries-dictionary}{Magyar}

```

Provide translations:

```

7081 \providetranslation{Glossary}{Sz\'ojegyz\'ek}
7082 \providetranslation{Acronyms}{Bet\H uszavak}
7083 \providetranslation{Notation (glossaries)}{Kifejez\'es}
7084 \providetranslation{Description (glossaries)}{Magyar\'azat}
7085 \providetranslation{Symbol (glossaries)}{Jel\'ol\'es}
7086 \providetranslation{Page List (glossaries)}{Oldalsz\'am}
7087 \providetranslation{Symbols (glossaries)}{Jelek}
7088 \providetranslation{Numbers (glossaries)}{Sz\'amjegyek}

```

6.12 Polish Dictionary

This is a dictionary file provided for use with the package.

```

7089 \ProvidesDictionary{glossaries-dictionary}{Polish}

```

Provide Polish translations:

```

7090 \providetranslation{Glossary}{S\lownik termin\'ow}
7091 \providetranslation{Acronyms}{Skr\'ot}
7092 \providetranslation{Notation (glossaries)}{Termin}
7093 \providetranslation{Description (glossaries)}{Opis}
7094 \providetranslation{Symbol (glossaries)}{Symbol}
7095 \providetranslation{Page List (glossaries)}{Strony}

```



```
7096 \providetranslation{Symbols (glossaries)}{Symbole}
7097 \providetranslation{Numbers (glossaries)}{Liczby}
```

6.13 Serbian Dictionary

This dictionary was provided by Zoran Filipovic.

```
7098 \ProvidesDictionary{glossaries-dictionary}{Serbian}
7099 \providetranslation{Glossary}{Mali re\vnik}
7100 \providetranslation{Acronyms}{Skra\'cenice}
7101 \providetranslation{Notation (glossaries)}{Oznaka}
7102 \providetranslation{Description (glossaries)}{Opis}
7103 \providetranslation{Symbol (glossaries)}{Simbol}
7104 \providetranslation{Page List (glossaries)}{Stranica}
7105 \providetranslation{Symbols (glossaries)}{Simboli}
7106 \providetranslation{Numbers (glossaries)}{Brojevi}
```

6.14 Spanish Dictionary

This is a dictionary file provided for use with the package.

```
7107 \ProvidesDictionary{glossaries-dictionary}{Spanish}
```

Provide Spanish translations:

```
7108 \providetranslation{Glossary}{Glosario}
7109 \providetranslation{Acronyms}{Siglas}
7110 \providetranslation{Notation (glossaries)}{Entrada}
7111 \providetranslation{Description (glossaries)}{Descripci\'on}
7112 \providetranslation{Symbol (glossaries)}{S\'mbolo}
7113 \providetranslation{Page List (glossaries)}{Lista de p\'aginas}
7114 \providetranslation{Symbols (glossaries)}{S\'mbolos}
7115 \providetranslation{Numbers (glossaries)}{N\'umeros}
```

Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.

Symbols	
\@glossarysec	<u>2</u>
\@glossaryseclabel	<u>3</u>
\@glossarysecstar	<u>3</u>
\@ACRlong	<u>229</u>
\@ACRshort	<u>228</u>
\@Acrlong	<u>229</u>
\@Acrshort	<u>228</u>
\@GLS@	<u>69</u> , <u>224</u>
\@GLSpl	<u>72</u>
\@GLSpl@	<u>227</u>
\@Gls@	<u>68</u> , <u>223</u>
\@Glspl@	<u>71</u> , <u>226</u>
\@acrlong	<u>229</u>
\@acrshort	<u>228</u>
\@addtoacronymlists	<u>9</u>
\@delimN	<u>137</u>
\@delimR	<u>137</u>
\@disable@onlypremakeg	<u>16</u>
\@disable@premakecs	<u>16</u>
\@disabled@gl saddx dy counters ..	<u>26</u>
\@do@seeglossary	<u>124</u>
\@do@wrglossary	<u>122</u> , <u>208</u>
\@glo@seeautonumberlist	<u>4</u>
\@glo@storeentry	<u>50</u>
\@glo@types	<u>37</u>
\@glossary	<u>121</u>
\@glossary@default@style	<u>3</u>
\@glossaryentryfield	<u>50</u> , <u>230</u>
\@glossarysection	<u>24</u>
\@glossarysubentryfield ..	<u>50</u> , <u>230</u>
\@gls	<u>67</u>
\@gls@	<u>67</u> , <u>222</u>
\@gls@@link	<u>56</u>
\@gls@addpredefinedattributes ..	<u>28</u>
\@gls@checkactual	<u>63</u>
\@gls@checkbar	<u>62</u>
\@gls@checkescactual	<u>61</u>
\@gls@checkescbar	<u>61</u>
\@gls@checkesclevel	<u>62</u>
\@gls@checkescquote	<u>60</u>
\@gls@checklevel	<u>63</u>
\@gls@checkmkidxchars	<u>59</u>
\@gls@checkquote	<u>60</u>
\@gls@codepage	<u>34</u>
\@gls@counterwithin	<u>6</u>
\@gls@escbsdq	<u>59</u>
\@gls@fixbraces	<u>124</u>
\@gls@getcounter	<u>39</u>
\@gls@getcounterprefix	<u>123</u>
\@gls@hypergroup	<u>166</u>
\@gls@ifinlist	<u>26</u>
\@gls@link	<u>56</u>
\@gls@loadlist	<u>4</u>
\@gls@loadlong	<u>4</u>
\@gls@loadsuper	<u>4</u>
\@gls@loadtree	<u>5</u>
\@gls@missingnumberlist	<u>44</u>
\@gls@noaccess	<u>216</u>
\@gls@onlypremakeg	<u>16</u>
\@gls@pl@	<u>225</u>
\@gls@renewglossary	<u>122</u>
\@gls@sanitizedesc	<u>11</u>
\@gls@sanitizename	<u>11</u>
\@gls@sanitizesort	<u>11</u>
\@gls@sanitizesymbol	<u>11</u>
\@gls@saveentrycounter	<u>57</u>
\@gls@setcounter	<u>39</u>
\@gls@setupsort@def	<u>7</u>
\@gls@setupsort@standard	<u>6</u>
\@gls@setupsort@use	<u>7</u>
\@gls@tmpb	<u>60</u>
\@gls@toc	<u>25</u>
\@gls@updatechecked	<u>60</u>
\@gls@xdy@Lclass@Alpha-page-numbers	<u>30</u>
\@gls@xdy@Lclass@Appendix-page-numbers	<u>30</u>
\@gls@xdy@Lclass@Roman-page-numbers	<u>29</u>
\@gls@xdy@Lclass@alpha-page-numbers	<u>29</u>
\@gls@xdy@Lclass@arabic-page-numbers	<u>29</u>
\@gls@xdy@Lclass@arabic-section-numbers	<u>30</u>

\@gls@xdy@Lclass@roman-page-numbers\ac	157		
.....	29	access (key)	214
\@gls@xdy@locationlist	29	accsupp package	214
\@gls@xdy@checkbackslash	64	\accsuppglossaryentryfield ..	230
\@gls@xdy@checkquote	64	\accsuppglossarysubentryfield	230
\@gls@Alphacompositor	20, 30	\Acf	157
\@gls@acronymlists	9	\acf	157
\@gls@defaultplural	44	\Acfp	157
\@gls@defaultsort	44	\acfp	157
\@gls@disp	73	\Acl	156
\@gls@firstletter	112	\acl	156
\@gls@hypernumber	137	\Aclp	157
\@gls@link	65	\aclp	156
\@gls@minrange	112	\Acp	157
\@gls@nextpages	130	\acp	157
\@gls@noname	43	\acrfootnote	145
\@gls@nonextpages	130	\ACRfull	141
\@gls@openfile	119	\Acrfull	141
\@gls@order	13	\acrfull	140, 141
\@gls@pl@	70	\acrfullformat	141
\@gls@target	65	\ACRfullpl	143
\@gls@widestname	204	\Acrfullpl	142
\@ist@filename	20	\acrfullpl	142
\@makeglossary	118	\acrlinkfootnote	145
\@newglossary	39	\acrlinkfullformat	141
\@newglossaryentryposthook ...	49	\ACRlong	101
\@newglossaryentryprehook	49	\Acrlong	101
\@nopostdesc	19	\acrlong	100
\@onlypremakeg	16	\ACRlongpl	103
\@p@glossarysection	24	\Acrlongpl	103
\@set@glo@numformat	58, 209	\acrlongpl	102
\@sgls	67, 73	\acrnameformat	143, 149
\@sgls@link	56	\acrnolinkfootnote	145
\@wrglossary	122	acronym (option)	9
\@xdy@main@language	14	\acronymfont .	143, 146, 149, 151, 152
\@xdy@attributelist	25	acronymlists (option)	10
\@xdy@attributes	25	\acronymname	17
\@xdy@language	34	\acronymtype	8, 139, 140
\@xdy@lettergroups	34	\acrpluralsuffix	140
\@xdy@locationclassorder	31	\ACRshort	98
\@xdy@locref	25	\Acrshort	97
\@xdy@requiredstyles	32	\acrshort	97
\@xdy@sortrules	31	\ACRshortpl	100
\@xdy@useralphabets	28	\Acrshortpl	99
\@xdy@userlocationdefs	30	\acrshortpl	98
\@xdy@userlocationnames	30	\Acs	156
		\acs	156
		\Acsp	156
		\acsp	156
A			
\Ac	157		

<code>\addglossarytocaptions</code>	18	<code>\delimN</code>	22 , 136
<code>\addto</code>	18	<code>\delimR</code>	22 , 136
<code>align (environment)</code>	57	<code>description (environment)</code>	170
<code>altlist (style)</code>	171	<code>description (key)</code>	40
<code>altlistgroup (style)</code>	171	<code>description (option)</code>	13
<code>altlisthypergroup (style)</code>	171	<code>descriptionaccess (key)</code>	215
<code>atlong4col (style)</code>	177	<code>\DescriptionDUANewAcronymDef</code>	147
<code>atlong4colborder (style)</code>	178	<code>\DescriptionFootnoteNewAcronymDef</code>	145 , 231
<code>atlong4colheader (style)</code>	177	<code>\descriptionname</code>	17
<code>atlong4colheaderborder (style)</code>	178	<code>\DescriptionNewAcronymDef</code>	148 , 232
<code>atlongragged4col (style)</code>	182	<code>descriptionplural (key)</code>	40
<code>atlongragged4colborder (style)</code>	183	<code>descriptionpluralaccess (key)</code>	215
<code>atlongragged4colheader (style)</code>	182	<code>dua (option)</code>	13
<code>atlongragged4colheaderborder</code> <code>(style)</code>	183	<code>\DUANewAcronymDef</code>	153
<code>\altnewglossary</code>	39	E	
<code>altsuper4col (style)</code>	192	<code>entrycounter (option)</code>	5
<code>altsuper4colborder (style)</code>	193	<code>entrycounterwithin (option)</code>	5
<code>altsuper4colheader (style)</code>	193	<code>\entryname</code>	17
<code>altsuper4colheaderborder (style)</code>	193	<code>environments:</code>	
<code>altsuperragged4col (style)</code>	197	<code>align</code>	57
<code>altsuperragged4colborder (style)</code>	198	<code>description</code>	170
<code>altsuperragged4colheader (style)</code>	198	<code>longtable</code>	4 , 158 , 173–183
<code>altsuperragged4colheaderborder</code> <code>(style)</code>	199	<code>multicols</code>	184
<code>alttree (style)</code>	204	<code>supertabular</code>	4 , 158 , 187–199
<code>alttreegroup (style)</code>	206	<code>theglossary</code>	22 , 133 , 135 , 185 , 186 , 201 , 202 , 204
<code>alttreehypergroup (style)</code>	207	<code>theindex</code>	199
<code>amsgen package</code>	2 , 55	<code>equation counter</code>	57 , 58
<code>\andname</code>	17	<code>etoolbox package</code>	2 , 163
<code>array package</code>	178 , 194	F	
<code>article class</code>	123	<code>file types</code>	
B		<code>.aux</code>	128
<code>babel package</code>	15 , 16 , 18 , 33 , 235	<code>.glo</code>	50
C		<code>.ist</code>	111 , 118
<code>\capitalisewords</code>	165	<code>.toc</code>	25
<code>compatible-2.07 (option)</code>	15	<code>.xdy</code>	20
<code>counter (key)</code>	42	<code>glo</code>	163
<code>counter (option)</code>	10	<code>\findrootlanguage</code>	32
<code>\CustomAcronymFields</code>	155	<code>first (key)</code>	41
<code>\CustomNewAcronymDef</code>	155	<code>firstaccess (key)</code>	214
D		<code>\firstacronymfont</code>	143
<code>\DeclareAcronymList</code>	9	<code>firstplural (key)</code>	41
<code>\DefaultNewAcronymDef</code> ...	144 , 231	<code>firstpluralaccess (key)</code>	215
<code>\defglsdisplay</code>	38 , 40 , 41 , 54 , 55	<code>footnote (option)</code>	13
<code>\defglsdisplayfirst</code>	38 , 40 , 41 , 54 , 55	<code>\FootnoteNewAcronymDef</code> ..	150 , 232
<code>\DefineAcronymSynonyms</code>	156	<code>\forallglossaries</code>	35
		<code>\forallglsentries</code>	35

`\forglsentries` [35](#)

G

`\glolinkprefix` [57](#)

`glossareentry counter` [131](#)

`glossaries package`
..... [34](#), [112](#), [158](#), [170](#), [207](#), [214](#)

`glossaries-accsupp package` [50](#), [214](#)

`\GlossariesWarning` [15](#)

`\GlossariesWarningNoLine` [15](#)

`\glossary` [38](#), [118](#), [121](#), [135](#)

`glossary counters:`

`glossaryentry` [130](#)

`glossarysubentry` [131](#)

`glossary keys:`

`access` [214](#)

`counter` [42](#)

`description` [40](#)

`descriptionaccess` [215](#)

`descriptionplural` [40](#)

`descriptionpluralaccess` .. [215](#)

`first` [41](#)

`firstaccess` [214](#)

`firstplural` [41](#)

`firstpluralaccess` [215](#)

`long` [43](#)

`longaccess` [215](#)

`longplural` [43](#)

`longpluralaccess` [215](#)

`name` [40](#)

`nonumberlist` [42](#)

`parent` [42](#)

`plural` [41](#)

`pluralaccess` [215](#)

`see` [42](#)

`short` [43](#)

`shortaccess` [215](#)

`shortplural` [43](#)

`shortpluralaccess` [215](#)

`sort` [40](#)

`symbol` [41](#)

`symbolaccess` [215](#)

`symbolplural` [41](#)

`symbolpluralaccess` [215](#)

`text` [40](#)

`textaccess` [214](#)

`type` [41](#)

`user1` [42](#)

`user2` [42](#)

`user3` [43](#)

`user4` [43](#)

`user5` [43](#)

`user6` [43](#)

`glossary package` [1](#), [139](#)

`glossary styles:`

`altlist` [171](#)

`altlistgroup` [171](#)

`altlisthypergroup` [171](#)

`altlong4col` [177](#), [178](#), [182](#)

`altlong4colborder` [178](#)

`altlong4colheader` [177](#)

`altlong4colheaderborder` .. [178](#)

`altlongragged4col` [182](#), [183](#)

`altlongragged4colborder` .. [183](#)

`altlongragged4colheader` .. [182](#)

`altlongragged4colheaderborder`
 [183](#)

`altsuper4col` [192](#), [193](#), [197](#)

`altsuper4colborder` [193](#)

`altsuper4colheader` [193](#)

`altsuper4colheaderborder` . [193](#)

`altsuperragged4col` ... [197](#)–[199](#)

`altsuperragged4colborder` . [198](#)

`altsuperragged4colheader` . [198](#)

`altsuperragged4colheaderborder`
 [199](#)

`alttree` [186](#), [204](#), [206](#)

`alttreegroup` [206](#), [207](#)

`alttreehypergroup` [207](#)

`index` [184](#), [199](#)–[201](#)

`indexgroup` [200](#), [201](#)

`indexhypergroup` [201](#)

`inline` [168](#)

`list` [3](#), [170](#)–[172](#)

`listdotted` [172](#)

`listgroup` [170](#), [171](#)

`listhypergroup` [171](#)

`long` [173](#), [174](#), [179](#)

`long3col` [174](#), [175](#)

`long3colborder` [175](#)

`long3colheader` [175](#)

`long3colheaderborder` [175](#)

`long4col` [176](#), [177](#)

`long4colborder` [177](#)

`long4colheader` [176](#)

`long4colheaderborder` [177](#)

`longborder` [174](#)

longheader	174	treehypergroup	202
longheaderborder	174	treenoname	185 , 203
longragged	179 , 180	treenonamegroup	203 , 204
longragged3col	180 , 181	treenonamehypergroup	204
longragged3colborder	181	glossary-hypernav package	111
longragged3colheader	181	glossary-list package	3 , 5 , 170
longragged3colheaderborder	181	glossary-long package	4 , 173 , 182 , 187 , 188
longraggedborder	179	glossary-longragged package	178
longraggedheader	180	glossary-mcols package	184
longraggedheaderborder	180	glossary-super package	
mcolalttree	186	4 , 173 , 187 , 194 , 197
mcolalttreegroup	186 , 187	glossary-superragged package	194
mcolalttreehypergroup	187	glossary-tree package	5 , 199
mcolindex	184	glossaryentry (counter)	130
mcolindexgroup	184	glossaryentry counter	5 , 131 , 132
mcolindexhypergroup	184	\glossaryentryfield	41 , 133 , 135 , 136
mcoltree	185	\glossaryentrynumber	130
mcoltreegroup	185	\glossaryentrynumbers	4 , 21 , 127 , 128
mcoltreehypergroup	185	\glossaryheader	133 , 135
mcoltreenoname	185 , 186	\glossarymark	5 , 23
mcoltreenonamegroup	186	\glossaryname	17 , 18
mcoltreenonamehypergroup	186	\glossarypostamble	22 , 135
sublistdotted	172	\glossarypreamble	22 , 135
super	188 , 189 , 195	\glossarysection	2 , 22 , 38
super3col	189 , 190	\glossarystyle	135 , 158
super3colborder	190	glossarysubentry (counter)	131
super3colheader	190	glossarysubentry counter	6 , 131 , 132
super3colheaderborder	190	\GLS	69
super4col	191 , 192	\Gls	68 , 71 , 163
super4colborder	192	\gls	2 , 41 , 53 , 54 , 66 , 69 , 70 , 74 , 76 , 77 , 79 , 80 , 82 , 83 , 85 , 86 , 88 , 89 , 91 , 92 , 94 , 95 , 132
super4colheader	191	\gls@checkisacronymlist	10
super4colheaderborder	192	\gls@codepage	14
superborder	188	\gls@docclearpage	24
superheader	189	\gls@hypergroup prerun	166
superheaderborder	189	\gls@level	44
superragged	194 – 196	\gls@save@numberlist	126
superragged3col	196 , 197	\gls@suffixF	21
superragged3colborder	196	\gls@suffixFF	21
superragged3colheader	197	\glsaccessdisplay	221
superragged3colheaderborder	197	\glsaccsupp	218
superraggedborder	195	\glsadd	53 , 110 , 135
superraggedheader	195	\glsadd options	
superraggedheaderborder	195	counter	110
superraggedright3colheaderborder	197	format	110 , 136
tree	185 , 201 – 204	\glsaddall	53 , 111
treegroup	185 , 202	\glsaddall options	
		types	110 , 111

<code>\GlsAddLetterGroup</code>	35	<code>\Glsentryfullpl</code>	109
<code>\GlsAddSortRule</code>	32	<code>\glentryfullpl</code>	108
<code>\GlsAddXdyAlphabet</code>	28	<code>\glentryitem</code>	132
<code>\GlsAddXdyAttribute</code>	27, 207	<code>\Glsentrylong</code>	108
<code>\GlsAddXdyCounters</code>	26, 208	<code>\glentrylong</code>	108
<code>\GlsAddXdyLocation</code>	30, 208	<code>\glentrylongaccess</code>	218
<code>\GlsAddXdyStyle</code>	32	<code>\Glsentrylongpl</code>	108
<code>\glscutprefix</code>	3	<code>\glentrylongpl</code>	108
<code>\glsclearpage</code>	25	<code>\glentrylongpluralaccess</code> ..	218
<code>\glsclosebrace</code>	111	<code>\Glsentryname</code>	104
<code>\glscpositor</code>	20, 30	<code>\glentryname</code>	104, 125
<code>\glscounter</code>	10, 39	<code>\glentrynumberlist</code>	109
<code>\glscdefaulttype</code>	8	<code>\Glsentryplural</code>	105
<code>\glscdefmain</code>	8	<code>\glentryplural</code>	105
<code>\GLSdesc</code>	83	<code>\glentrypluralaccess</code>	217
<code>\Glsdesc</code>	82	<code>\Glsentryshort</code>	108
<code>\glscdesc</code>	82, 83	<code>\glentryshort</code>	107
<code>\GLSdescplural</code>	84	<code>\glentryshortaccess</code>	218
<code>\Glsdescplural</code>	84	<code>\Glsentryshortpl</code>	108
<code>\glscdescplural</code>	83, 84	<code>\glentryshortpl</code>	108
<code>\glscdescriptionaccessdisplay</code>	220	<code>\glentryshortpluralaccess</code> ..	218
<code>\glscdescriptionpluralaccessdisplay</code>	220	<code>\glentrysort</code>	106
<code>\glscdescwidth</code>	173, 178, 187, 194	<code>\Glsentrysymbol</code>	105
<code>\glscdisablehyper</code>	66	<code>\glentrysymbol</code>	105
<code>\glscdisp</code>	73	<code>\glentrysymbolaccess</code>	218
<code>\glscdisplay</code>	10, 40, 41, 53, 54, 66	<code>\Glsentrysymbolplural</code>	106
<code>\glscdisplayfirst</code> ..	40, 41, 53, 54, 66	<code>\glentrysymbolplural</code>	105
<code>\glscdisplaynumberlist</code>	109	<code>\glentrysymbolpluralaccess</code> ..	218
<code>\glscdoifexists</code>	36	<code>\Glsentrytext</code>	105
<code>\glscdoifnoexists</code>	36	<code>\glentrytext</code>	105, 125
<code>\glscenablehyper</code>	66	<code>\glentrytextaccess</code>	217
<code>\glscentryaccess</code>	217	<code>\glentrytype</code>	106
<code>\glscentrycounter</code>	57	<code>\Glsentryuseri</code>	106
<code>\glscentrycounterlabel</code>	132	<code>\glentryuseri</code>	106
<code>\Glsentrydesc</code>	104	<code>\Glsentryuserii</code>	107
<code>\glscentrydesc</code>	10, 104	<code>\glscentryuserii</code>	107
<code>\glscentrydescaccess</code>	218	<code>\glscentryuseriii</code>	107
<code>\Glsentrydescplural</code>	105	<code>\glscentryuseriii</code>	107
<code>\glscentrydescplural</code>	105	<code>\Glsentryuseriv</code>	107
<code>\glscentrydescpluralaccess</code> ..	218	<code>\glscentryuseriv</code>	107
<code>\Glsentryfirst</code>	106	<code>\Glsentryuserv</code>	107
<code>\glscentryfirst</code>	106	<code>\glscentryuserv</code>	107
<code>\glscentryfirstaccess</code>	217	<code>\glscentryuservi</code>	107
<code>\Glsentryfirstplural</code>	106	<code>\GLSfirst</code>	77
<code>\glscentryfirstplural</code>	106	<code>\Glsfirst</code>	76, 77
<code>\glscentryfirstpluralaccess</code> ..	217	<code>\glscfirst</code>	76
<code>\Glsentryfull</code>	108	<code>\glscfirstaccessdisplay</code>	219
<code>\glscentryfull</code>	108	<code>\GLSfirstplural</code>	80

<code>\Glsfirstplural</code>	79	<code>\glsnavhyperlink</code>	166
<code>\glsfirstplural</code>	79, 80	<code>\glsnavhypertarget</code>	166
<code>\glsfirstpluralaccessdisplay</code>	220	<code>\glsnavigation</code>	167
<code>\glsgetgrouplabel</code>	135	<code>\glsnextpages</code>	130
<code>\glsgetgrouptitle</code>	111, 134	<code>\glsnonextpages</code>	130
<code>\glsgroupheading</code>	134, 135	<code>\glsnoxindywarning</code>	25
<code>\glsgroupskip</code>	134, 135, 170	<code>\glsnumberformat</code>	21
<code>\glshyperlink</code>	110	<code>\glsnumbersgroupname</code> ..	17, 111, 134
<code>\glshypernavsep</code>	167	<code>\glsnumlistlastsep</code>	110
<code>\glshypernumber</code>	21, 136	<code>\glsnumlistsep</code>	110
<code>\glsIfListOfAcronyms</code>	9	<code>\glsopenbrace</code>	111
<code>\glsinlinedescformat</code>	169	<code>\glsorder</code>	13
<code>\glsinlinedopostchild</code> ...	168, 169	<code>\glspagelistwidth</code>	173, 179, 188, 194
<code>\glsinlineemptydescformat</code> ...	169	<code>\glspar</code>	19
<code>\glsinlinenameformat</code>	169	<code>\GLSpl</code>	72
<code>\glsinlineparentchildseparator</code>		<code>\Glspl</code>	71, 163
.....	169	<code>\glspl</code>	53, 54, 70–72
<code>\glsinlinepostchild</code>	169	<code>\GLSplural</code>	78
<code>\glsinlineseparator</code>	169	<code>\Glsplural</code>	78
<code>\glsinlinesubdescformat</code>	169	<code>\glsplural</code>	77, 78
<code>\glsinlinesubnameformat</code>	169	<code>\glspluralaccessdisplay</code>	219
<code>\glsinlinesubseparator</code>	169	<code>\glspluralsuffix</code>	17, 41
<code>\glskeylisttok</code>	143	<code>\glspostdescription</code>	5
<code>\glslabeltok</code>	143	<code>\glspostinline</code>	169
<code>\glslink</code>	53, 56, 66, 110, 135, 136	<code>\glsquote</code>	112
<code>\glslink options</code>		<code>\glsrefentry</code>	132
counter	55, 66, 163	<code>\glsreset</code>	52
format	55, 66, 136	<code>\glsresetall</code>	52
hyper	55, 66	<code>\glsresetentrylist</code>	130
<code>\glslistdottedwidth</code>	172	<code>\glsresetsubentrycounter</code>	131
<code>\glslocalreset</code>	52	<code>\glssee</code>	125
<code>\glslocalresetall</code>	52	<code>\glsseeformat</code>	114, 125
<code>\glslocalunset</code>	52	<code>\glsseeitem</code>	125
<code>\glslocalunsetall</code>	53	<code>\glsseeitemformat</code>	125
<code>\glslongaccessdisplay</code>	221	<code>\glsseelastsep</code>	125
<code>\glslongaccesskey</code>	234	<code>\glsseelist</code>	125
<code>\glslongkey</code>	140	<code>\glsseesep</code>	125
<code>\glslongpluralaccessdisplay</code> .	221	<code>\glsSetAlphaCompositor</code>	21
<code>\glslongpluralaccesskey</code>	234	<code>\glsSetCompositor</code>	20
<code>\glslongpluralkey</code>	140	<code>\glsSetSuffixF</code>	21
<code>\glslongtok</code>	143	<code>\glsSetSuffixFF</code>	21
<code>\glsmakefirsttuc</code>	164	<code>\glssettoctitle</code>	17
<code>\glsmcols</code>	184	<code>\glssetwidest</code>	204
<code>\glsmoveentry</code>	49	<code>\GlsSetXdyCodePage</code>	34
<code>\GLSname</code>	81	<code>\GlsSetXdyFirstLetterAfterDigits</code>	
<code>\Glsname</code>	81	112
<code>\glsname</code>	80, 81	<code>\GlsSetXdyLanguage</code>	34
<code>\glsnameaccessdisplay</code>	219	<code>\GlsSetXdyLocationClassOrder</code> .	31
<code>\glsnamefont</code>	136	<code>\GlsSetXdyMinRangeLength</code>	112

\GlsSetXdyStyles	32	\Glsuservi	96
\glsshortaccessdisplay	221	\glssuservi	95, 96
\glsshortaccesskey	234	\glswrite	120
\glsshortkey	140	\glswritefiles	120
\glsshortpluralaccessdisplay	221		
\glsshortpluralaccesskey	234	H	
\glsshortpluralkey	140	\hyperbf	138
\glsshorttok	143	\hyperemph	139
\glssortnumberfmt	6	hyperfirst (option)	13
\glssteppentry	131	\hyperit	138
\glssteppsubentry	131	\hyperlink	65
\glssubentrycounterlabel	132	\hypermd	138
\glssubentryitem	132	\hyperpage	136
\GLSsymbol	86	hyperref package	123, 126, 136, 163
\Glsymbol	85	\hyperrm	138
\glssymbol	85, 86	\hypersc	139
\glssymbolaccessdisplay	220	\hypersf	138
\glssymbolnav	167	\hypersl	138
\GLSsymbolplural	87	\hypertarget	65
\Glsymbolplural	87	\hypertt	138
\glssymbolplural	86, 87	\hyperup	138
\glssymbolpluralaccessdisplay	220		
\glssymbolsgroupname ..	17, 111, 134	I	
\glstarget	133	\if@glsisacronymlist	10
\GLStext	75	\ifglossaryexists	36
\Glstext	75	\ifglstryexists	36
\glstext	74	\ifglshaschildren	37
\glstextaccessdisplay	219	\ifglshasparent	37
\glstextformat	53, 54	\ifglused	36, 52
\glstreeindent	202, 203	\ifglxindy	14
\glset	52	index (style)	199
\glsetall	53	indexgroup (style)	200
\GLSuseri	89	indexhypergroup (style)	201
\Glsuseri	88	indexonlyfirst (option)	12
\gluseri	88, 89	inline (style)	168
\GLSuserii	90	\inputencodingname	14
\Glsuserii	90	\istfile	120
\gluserii	89, 90	\istfilename	20
\GLSuseriii	92	\item	136, 170, 199, 200
\Glsuseriii	91		
\gluseriii	91, 92	L	
\GLSuseriv	93	\language	32
\Glsuseriv	93	link text	53
\gluseriv	92, 93	list (style)	170
\GLSuserv	95	listdotted (style)	172
\Glsuserv	94	listgroup (style)	170
\gluserv	94, 95	listhypergroup (style)	171
\GLSuservi	96	\loadglsentries	8, 53
		long (key)	43
		long (style)	173

long3col (style)	174	mcoltreegroup (style)	185
long3colborder (style)	175	mcoltreehypergroup (style)	185
long3colheader (style)	175	mcoltreenoname (style)	185
long3colheaderborder (style) ...	175	mcoltreenonamegroup (style) ...	186
long4col (style)	176	mcoltreenonamehypergroup (style)	186
long4colborder (style)	177	memoir class	121
long4colheader (style)	176	mfistuc package	1
long4colheaderborder (style) ...	177	multicol package	184
longaccess (key)	215	multicols (environment)	184
longborder (style)	174		
longheader (style)	174	N	
longheaderborder (style)	174	name (key)	40
longplural (key)	43	\newacronym	13, 43, 53, 139, 140
longpluralaccess (key)	215	\newacronymhook	143
longragged (style)	179	\newglossary .	10, 38, 39, 118, 119, 129
longragged3col (style)	180	\newglossaryentry ...	10, 44, 53, 140
longragged3colborder (style) ...	181	\newglossaryentry options	
longragged3colheader (style) ...	181	access	216, 217
longragged3colheaderborder		counter	42
(style)	181	description	10, 11,
longraggedborder (style)	179		13, 40, 44, 54, 82, 104, 140, 150, 215
longraggedheader (style)	180	descriptionaccess	218, 220
longraggedheaderborder (style) .	180	descriptionplural	83, 215
longtable (environment)		descriptionpluralaccess ...	218, 220
	4, 158, 173–183	first	41,
longtable package	173, 178		47, 54, 66, 76, 106, 149, 152, 214
		firstaccess	217, 219
M		firstplural ...	41, 47, 54, 79, 106, 215
\makefirstuc	163	firstpluralaccess	217, 220
makeglossaries		format	113
...	13, 20, 32, 34, 38, 119, 126, 128	long	108, 215
\makeglossaries		longaccess	218, 221
...	16, 20, 21, 37, 39, 119, 120	longplural	108, 215
\makeglossary	120	longpluralaccess	218, 221
makeindex		name	10,
...	14, 17, 20, 22, 37–40, 51, 58,		11, 40, 43, 44, 80, 104, 125, 214
	61, 111, 114, 116, 118, 123, 134, 209	nonumberlist	42
delim_n	22	parent	42, 44
delim_r	22	plural	41, 47, 54, 77, 215
page_compositor	20	pluralaccess	217, 219
special characters	59, 60, 111	see	4, 42
\MakeUppercase	5	short	107, 215
mcolalttree (style)	186	shortaccess	218, 221
mcolalttreegroup (style)	186	shortplural	108, 215
mcolalttreehypergroup (style) ..	187	shortpluralaccess	218, 221
mcolindex (style)	184	sort	10, 12, 40, 106, 134
mcolindexgroup (style)	184	symbol	
mcolindexhypergroup (style) ...	184		10, 11, 40, 41, 54, 85, 105, 146,
mcoltree (style)	185		147, 149, 152, 176, 191, 214–216

symbolaccess	218, 220
symbolplural	86, 215
symbolpluralaccess	218, 220
text	40, 41, 54, 66, 74, 105, 146, 149, 214
textaccess	217, 219
type	8, 41, 53, 106
user1	88, 106, 216
user2	89, 107
user3	91, 107
user4	92, 107
user5	94, 107
user6	95, 107, 216
\newglossarystyle	135
ngerman package	33
nogroupskip (option)	5
\noist	118, 163, 214
nolist (option)	5
nolong (option)	4
nonumberlist (key)	42
nonumberlist (option)	4
\nopostdesc	19
nopostdot (option)	5
nostyles (option)	5
nosuper (option)	4
notree (option)	5
numberedsection (option)	3
numberline (option)	2
O	
\oldacronym	139
order (option)	13
P	
package options:	
acronym	8, 9, 17, 128, 139, 140
acronym	9
acronymlists	10
compatible-2.07	15
counter	10
counter	10
description	149
description	13
dua	148, 149
dua	13
entrycounter	130
true	5
entrycounter	5
entrycounterwithin	5
footnote	67–70, 72–74, 146, 148–150, 222–227
footnote	13
hyperfirst	
false	67–70, 72–74
hyperfirst	13
indexonlyfirst	12
makeindex	114, 163
nogroupskip	5
nolist	158
nolist	5
nolong	158, 173
nolong	4
nomain	8
nonumberlist	4
nonumberlist	4
nopostdot	5
nostyles	5
nosuper	158
nosuper	4
notree	158
notree	5
numberedsection	3
numberline	2
numberline	2
order	13
sanitize	11, 12, 40, 104, 105
sanitize	12
savenumberlist	4
savewrites	14
false	118
true	120
savewrites	14
section	2, 23
section	2
seeautonumberlist	4
shotcuts	13
smallcaps	13
smaller	13
sort	
def	6
standard	6
use	6
sort	6
style	3, 158
style	3
subentrycounter	131
subentrycounter	6
toc	2

true	2	\SetDescriptionDUAACronymStyle	147
toc	2	\SetDescriptionFootnoteAcronymDisplayStyle	145
translate	12	\SetDescriptionFootnoteAcronymStyle	146
ucmark	5	\SetDUADisplayStyle	153
xindy	14 , 114 , 163	\SetDUASyle	153
\pagelistname	17	\setentrycounter	135
parent (key)	42	\SetFootnoteAcronymDisplayStyle	149
\phantomsection	22 , 24	\SetFootnoteAcronymStyle	150
plural (key)	41	\setglossarysection	24
pluralaccess (key)	215	\SetSmallAcronymDisplayStyle	151
polyglossia package	15 , 18	\SetSmallAcronymStyle	152
\printglossaries	8 , 22 , 37 , 39 , 120 , 126 , 129 , 165	\setStyleFile	19 , 20
\printglossary	22 , 37 , 120 , 126 , 128 , 129 , 165 , 166	short (key)	43
\printglossary options		shortaccess (key)	215
nonumberlist	130	shortplural (key)	43
numberedsection	129	shortpluralaccess (key)	215
style	129	shotcuts (option)	13
title	129	\showacronymlists	161
toctitle	129	\showglocounter	159
type	8 , 126 , 129	\showglodesc	160
\protect	11	\showglodescaccess	235
R			
\renewglossarystyle	136	\showglodescplural	161
\roman	29	\showglodescpluralaccess	235
\rootlanguage name	33 , 34	\showglofirst	159
S			
sanitize (option)	12	\showglofirstaccess	234
savenumberlist (option)	4	\showglofirststpl	159
savewrites (option)	14	\showglofirststplpluralaccess ...	234
section (option)	2	\showgloflag	161
see (key)	42	\showgloindex	161
seeautonumberlist (option)	4	\showglolevel	158
\seename	17	\showglolongaccess	235
\SetAcronymLists	10	\showglolongpluralaccess	235
\SetAcronymStyle	9 , 154	\showgloname	160
\SetCustomDisplayStyle	155	\showglonameaccess	234
\SetCustomStyle	155	\showgloparent	158
\SetDefaultAcronymDisplayStyle	144	\showgloplural	159
\SetDefaultAcronymStyle	144	\showglopluralaccess	234
\SetDescriptionAcronymDisplayStyle	148	\showgloshortaccess	235
\SetDescriptionAcronymStyle .	149	\showgloshortpluralaccess ...	235
\SetDescriptionDUAACronymDisplayStyle	147	\showglosort	161
		\showglossaries	162
		\showglossarycounter	162
		\showglossaryentries	162
		\showglossaryin	162
		\showglossaryout	162

<code>\showglossarytitle</code>	162	<code>symbol (key)</code>	41
<code>\showglosymbol</code>	161	<code>symbolaccess (key)</code>	215
<code>\showglosymbolaccess</code>	234	<code>\symbolname</code>	17
<code>\showglosymbolplural</code>	161	<code>symbolplural (key)</code>	41
<code>\showglosymbolpluralaccess</code> ..	235	<code>symbolpluralaccess (key)</code>	215
<code>\showglotext</code>	158	T	
<code>\showglotextaccess</code>	234	<code>text (key)</code>	40
<code>\showglotype</code>	159	<code>textaccess (key)</code>	214
<code>\showglouserii</code>	159	<code>\theequation</code>	123
<code>\showglouseriii</code>	160	<code>theglossary (environment)</code> ...	22 , 133 , 135 , 185 , 186 , 201 , 202 , 204
<code>\showglouseriii</code>	160	<code>\theHequation</code>	123
<code>\showglouseriv</code>	160	<code>theindex (environment)</code>	199
<code>\showglouserv</code>	160	<code>toc (option)</code>	2
<code>\showglouservi</code>	160	<code>\translate</code>	18
<code>smallcaps (option)</code>	13	<code>translate (option)</code>	12
<code>smaller (option)</code>	13	<code>translator package</code>	
<code>\SmallNewAcronymDef</code>	151 , 233	15 , 18 , 126 , 235 , 245 – 248
<code>sort (key)</code>	40	<code>tree (style)</code>	201
<code>sort (option)</code>	6	<code>treegroup (style)</code>	202
<code>style (option)</code>	3	<code>treehypergroup (style)</code>	202
<code>subentrycounter (option)</code>	6	<code>treenoname (style)</code>	203
<code>\subitem</code>	200	<code>treenonamegroup (style)</code>	203
<code>sublistdotted (style)</code>	172	<code>treenonamehypergroup (style)</code> ...	204
<code>\subsubitem</code>	200	<code>type (key)</code>	41
<code>super (style)</code>	188	U	
<code>super3col (style)</code>	189	<code>ucmark (option)</code>	5
<code>super3colborder (style)</code>	190	<code>user1 (key)</code>	42
<code>super3colheader (style)</code>	190	<code>user2 (key)</code>	42
<code>super3colheaderborder (style)</code> ..	190	<code>user3 (key)</code>	43
<code>super4col (style)</code>	191	<code>user4 (key)</code>	43
<code>super4colborder (style)</code>	192	<code>user5 (key)</code>	43
<code>super4colheader (style)</code>	191	<code>user6 (key)</code>	43
<code>super4colheaderborder (style)</code> ..	192	W	
<code>superborder (style)</code>	188	<code>\warn@nomakeglossaries</code>	119
<code>superheader (style)</code>	189	<code>\warn@noprintglossary</code>	126
<code>superheaderborder (style)</code>	189	<code>\writeist</code> .	20 , 26 , 27 , 31 , 112 , 208 , 209
<code>superragged (style)</code>	194	X	
<code>superragged3col (style)</code>	196	<code>\xcapitalisewords</code>	165
<code>superragged3colborder (style)</code> ..	196	<code>\xglsaccsupp</code>	219
<code>superragged3colheader (style)</code> ..	197	<code>xindy</code>	14 , 20 , 25 , 28 , 30 , 32 , 34 , 35 , 50 , 51 , 64 , 112 , 114 , 123 , 126 , 128 , 134 , 163 , 209
<code>superraggedborder (style)</code>	195	<code>\xmakefirstuc</code>	165
<code>superraggedheader (style)</code>	195	<code>xspace package</code>	2 , 139
<code>superraggedheaderborder (style)</code> ..	195		
<code>superraggedright3colheaderborder</code> <code>(style)</code>	197		
<code>supertabular (environment)</code>			
.....	4 , 158 , 187 – 199		
<code>supertabular package</code> ...	4 , 158 , 187 , 194		