

Documented Code For glossaries v3.02

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/>

2012-05-21

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 v3.02: L^AT_EX2e Package to Assist Generating Glossaries”.

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

glossaries.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	17
1.4 Xindy	26
1.5 Loops and conditionals	36
1.6 Defining new glossaries	38
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	54
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	111
1.12 Creating associated files	112
1.13 Writing information to associated files	121
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	158
1.19 Debugging Commands	159
1.20 Compatibility with version 2.07 and below	163
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)	168
3.3 List Style (glossary-list.sty)	170
3.4 Glossary Styles using longtable (the glossary-long package)	173
3.5 Glossary Styles using longtable (the glossary-longragged package)	178
3.6 Glossary Styles using multicol (glossary-mcols.sty)	184
3.7 Glossary Styles using supertabular environment (glossary-super package)	187
3.8 Glossary Styles using supertabular environment (glossary-superragged package)	194
3.9 Tree Styles (glossary-tree.sty)	199
4 glossaries-compatible-207	207

5 Accessibility Support (glossaries-accsupp Code)	214
5.1 Defining Replacement Text	214
5.2 Accessing Replacement Text	217
5.3 Displaying the Glossary	230
5.4 Acronyms	230
5.5 Debugging Commands	234
6 Multi-Lingual Support	235
6.1 Babel Captions	235
6.2 Polyglossia Captions	241
6.3 Brazilian Dictionary	245
6.4 Danish Dictionary	245
6.5 Dutch Dictionary	245
6.6 English Dictionary	245
6.7 French Dictionary	246
6.8 German Dictionary	246
6.9 Irish Dictionary	246
6.10 Italian Dictionary	247
6.11 Magyar Dictionary	247
6.12 Polish Dictionary	247
6.13 Serbian Dictionary	248
6.14 Spanish Dictionary	248
Index	249

1 Main Package Code

1.1 Package Definition

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

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries}[2012/05/21 v3.02 (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 }

```

`ucmark` Boolean option to determine whether or not to use `\MakeUppercase` in definition of `\glossarymark`

```

66 \define@boolkey{glossaries.sty}[gls]{ucmark}[true]{}
67 \glsucmarkfalse

```

entrycounter Defines a counter that can be used in the standard glossary styles to number each (main) entry. If true, this will define a counter called glossaryentry.

```

68 \define@boolkey{glossaries.sty}[gls]{entrycounter}[true]{}
69 \glsentrycounterfalse

```

entrycounterwithin This option can be used to set a parent counter for glossaryentry. This option automatically sets entrycounter=true.

```

70 \define@key{glossaries.sty}{counterwithin}{%
71   \renewcommand*{\@gls@counterwithin}{#1}%
72   \glsentrycountertrue
73 }

```

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

```

74 \newcommand*{\@gls@counterwithin}{}

```

subentrycounter Define a counter that can be used in the standard glossary styles to number each level 1 entry. If true, this will define a counter called glossarysubentry.

```

75 \define@boolkey{glossaries.sty}[gls]{subentrycounter}[true]{}
76 \glssubentrycounterfalse

```

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

```

77 \define@choicekey{glossaries.sty}{sort}{standard,def,use}{%
78   \csname @gls@setupsort@#1\endcsname
79 }

```

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

```

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

```

Store entry information when it's defined.

```

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

```

No count register required for standard sort.

```

82   \def\@gls@defsortcount##1{%

```

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

```

83     \def\@gls@defsort##1##2{%
84       \ifx\@glo@sort\@glsdefaultsort
85         \let\@glo@sort\@glo@name
86       \fi
87       \@onelevel@sanitize\@glo@sort
88       \expandafter\protected@xdef\csname glo@##2@sort\endcsname{\@glo@sort}%
89     }%

```

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

```

90   \def\@gls@setsort##1{%
91 }

```

Set standard sort as the default:

```
92 \@gls@setupsort@standard
```

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

```
93 \newcommand*\glssortnumberfmt[1]{%
94   \ifnum#1<100000 0\fi
95   \ifnum#1<10000 0\fi
96   \ifnum#1<1000 0\fi
97   \ifnum#1<100 0\fi
98   \ifnum#1<10 0\fi
99   \number#1%
100 }
```

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

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

Store entry information when it's defined.

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

Defined count register associated with the glossary.

```
103   \def\@gls@defsortcount##1{%
104     \expandafter\global
105     \expandafter\newcount\csname glossary@##1@sortcount\endcsname
106   }%
```

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

```
107   \def\@gls@defsort##1##2{%
108     \expandafter\global\expandafter
109     \advance\csname glossary@##1@sortcount\endcsname by 1\relax
110     \expandafter\protected@xdef\csname glo@##2@sort\endcsname{%
111       \expandafter\glssortnumberfmt
112       {\csname glossary@##1@sortcount\endcsname}}%
113   }%
```

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

```
114   \def\@gls@setsort##1{%
115 }
```

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

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

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

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

Defined count register associated with the glossary.

```
118   \def\@gls@defsortcount##1{%
119     \expandafter\global
120     \expandafter\newcount\csname glossary@##1@sortcount\endcsname
121   }%
```


Initialise the sort key to empty.

```
122 \def\@gls@defsort##1##2{%  
123   \expandafter\gdef\csname glo@##2@sort\endcsname{%  
124   }%
```

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

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

Get the parent, if one exists

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

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

```
127   \ifx\@glo@parent\@empty  
128   \else  
129     \expandafter\@gls@setsort\expandafter{\@glo@parent}%  
130   \fi
```

Set index information for this entry

```
131   \edef\@glo@type{\csname glo@##1@type\endcsname}%  
132   \edef\@gls@tmp{\csname glo@##1@sort\endcsname}%  
133   \ifx\@gls@tmp\@empty  
134     \expandafter\global\expandafter  
135       \advance\csname glossary@\@glo@type @sortcount\endcsname by 1\relax  
136     \expandafter\protected@xdef\csname glo@##1@sort\endcsname{%  
137       \expandafter\glssortnumberfmt  
138       {\csname glossary@\@glo@type @sortcount\endcsname}}%  
139     \@glo@storeentry{##1}%  
140   \fi  
141 }%  
142 }
```

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

```
143 \newcommand*\glsdefmain{%  
144   \newglossary{main}{gls}{glo}{\glossaryname}%  
145 }
```

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`

```
146 \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`

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

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

```
148 \DeclareOptionX{nomain}{%
149   \let\glsdefaulttype\relax
150   \renewcommand*{\glsdefmain}{}%
151 }
```

`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.

```
152 \define@boolkey{glossaries.sty}[gls]{acronym}[true]{%
153   \DeclareAcronymList{acronym}%
154 }
```

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

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

`\@addtoacronymlists`

```
156 \newcommand*{\@addtoacronymlists}[1]{%
157   \ifx\@glsacronymlists\@empty
158     \protected@xdef\@glsacronymlists{#1}%
159   \else
160     \protected@xdef\@glsacronymlists{\@glsacronymlists,#1}%
161   \fi
162 }
```

`\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.)

```
163 \newcommand*{\DeclareAcronymList}[1]{%
164   \glsIfListOfAcronyms{#1}{\@addtoacronymlists{#1}}%
165 }
```

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

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

```
166 \newcommand{\glsIfListOfAcronyms}[1]{%
167   \edef\@do@glis@islistofacronyms{%
168     \noexpand\@glis@islistofacronyms{#1}{\@glsacronymlists}}%
169   \@do@glis@islistofacronyms
170 }
```

Internal command requires label and list to be expanded:

```
171 \newcommand{\@gls@islistofacronyms}[4]{%
172   \def\gls@islistofacronyms##1,#1,##2\end@gls@islistofacronyms{%
173     \def\@before{##1}\def\@after{##2}}%
174   \gls@islistofacronyms,#2,#1,\@nil\end@gls@islistofacronyms
175   \ifx\@after\@nnil
```

Not found

```
176     #4%
177   \else
```

Found

```
178     #3%
179   \fi
180 }
```

`\if@glssisacronymlist` Convenient boolean.

```
181 \newif\if@glssisacronymlist
```

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

```
182 \newcommand*\@gls@checkisacronymlist[1]{%
183   \glsIfListOfAcronyms{#1}%
184   {\@glssisacronymlisttrue}{\@glssisacronymlistfalse}%
185 }
```

`\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.)

```
186 \newcommand*\SetAcronymLists[1]{%
187   \renewcommand*\@glsacronymlists{#1}%
188 }
```

`acronymlists`

```
189 \define@key{glossaries.sty}{acronymlists}{%
190   \@addtoacronymlists{#1}%
191 }
```

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`

```
192 \newcommand{\glscounter}{page}
```

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

```
193 \define@key{glossaries.sty}{counter}{%
194   \renewcommand*\glscounter{#1}%
195 }
```

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
```

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

```
\@gls@sanitizename
```

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

```
\@gls@sanitizesymbol
```

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

(There is no equivalent for the sort key, since that is only provided for the benefit of `makeindex` or `xindy`, and so will always be sanitized.)

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.

```
199 \define@boolkey[gls]{sanitize}{description}[true]{%
200 \ifgls@sanitize@description
201   \renewcommand*{\@gls@sanitizedesc}{\@onelevel@sanitize\@glo@desc}%
202 \else
203   \renewcommand*{\@gls@sanitizedesc}{}%
204 \fi
205 }
```

Similarly for the name key:

```
206 \define@boolkey[gls]{sanitize}{name}[true]{%
```

```

207 \ifgls@sanitize@name
208   \renewcommand*{\@gls@sanitizename}{\@onelevel@sanitize\@glo@name}%
209 \else
210   \renewcommand*{\@gls@sanitizename}{}%
211 \fi}

```

and for the symbol key:

```

212 \define@boolkey[gls]{sanitize}{symbol}[true]{%
213 \ifgls@sanitize@symbol
214   \renewcommand*{\@gls@sanitizesymbol}{}%
215 \@onelevel@sanitize\@glo@symbol}%
216 \else
217   \renewcommand*{\@gls@sanitizesymbol}{}%
218 \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:

```

219 \define@key{glossaries.sty}{sanitize}[description=true,symbol=true,
220 name=true]{%
221 \ifthenelse{\equal{#1}{none}}{}%
222 \renewcommand*{\@gls@sanitizedesc}{}%
223 \renewcommand*{\@gls@sanitizename}{}%
224 \renewcommand*{\@gls@sanitizesymbol}{}%
225 }{\setkeys[gls]{sanitize}{#1}}%
226 }

```

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

```

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

```

Set the default value:

```

228 \glstranslatefalse
229 \@ifpackageloaded{translator}%
230   {\glstranslatetrue}%
231   {%
232     \@ifpackageloaded{polyglossia}%
233       {\glstranslatetrue}%
234       {%
235         \@ifpackageloaded{babel}{\glstranslatetrue}{}%
236       }%
237 }

```

indexonlyfirst Set whether to only index on first use.

```

238 \define@boolkey{glossaries.sty}[gls]{indexonlyfirst}[true]{}
239 \glsindexonlyfirstfalse

```

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

```

240 \define@boolkey{glossaries.sty}[gls]{hyperfirst}[true]{}
241 \glshyperfirsttrue

```

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

```

242 \define@boolkey{glossaries.sty}[glsacr]{footnote}[true]{%
243 \ifthenelse{\boolean{glsacrdescription}}{}}%
244 {\renewcommand*{\@gls@sanitizedesc}{}}%
245 }

```

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

```

246 \define@boolkey{glossaries.sty}[glsacr]{description}[true]{%
247 \renewcommand*{\@gls@sanitizesymbol}{}}%
248 }

```

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

```

249 \define@boolkey{glossaries.sty}[glsacr]{smallcaps}[true]{%
250 \renewcommand*{\@gls@sanitizesymbol}{}}%
251 }

```

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

```

252 \define@boolkey{glossaries.sty}[glsacr]{smaller}[true]{%
253 \renewcommand*{\@gls@sanitizesymbol}{}}%
254 }

```

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

```

255 \define@boolkey{glossaries.sty}[glsacr]{dua}[true]{%
256 \renewcommand*{\@gls@sanitizesymbol}{}}%
257 }

```

shortcuts Define acronym shortcuts.

```

258 \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.

```

259 \newcommand*\glsorder{word}

```

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

```

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

```

order

```

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

```

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

```

263 \newif\ifglxindy

```

The default is `makeindex`:

```
264 \glxindyfalse
```

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

```
265 \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 `glnumbers` determines whether to automatically add the `glnumbers` letter group.

```
266 \define@boolkey[gl]{xindy}{glnumbers}[true]{}
```

```
267 \glx@xindy@glnumberstrue
```

`\@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.)

```
268 \def\@xdy@main@language{\rootlanguage}%
```

Define key to set the language

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

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

```
270 \ifcsundef{inputencodingname}{%
```

```
271 \def\glscodepage{}}{%
```

```
272 \def\glscodepage{\inputencodingname}
```

```
273 }
```

Define a key to set the code page.

```
274 \define@key[gl]{xindy}{codepage}{\def\glscodepage{#1}}
```

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

```
275 \define@key{glossaries.sty}{xindy}[]{%
```

```
276 \glxindytrue
```

```
277 \setkeys[gl]{xindy}{#1}%
```

```
278 }
```

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

```
279 \define@boolkey{glossaries.sty}[gl]{savewrites}[true]{}
```

Set default:

```
280 \glssavewritesfalse
```

`\GlossariesWarning` Prints a warning message.

```
281 \newcommand*\GlossariesWarning[1]{%
```

```
282 \PackageWarning{glossaries}{#1}%
```

```
283 }
```

sariesWarningNoLine Prints a warning message without the line number.

```
284 \newcommand*{\GlossariesWarningNoLine}[1]{%
285   \PackageWarningNoLine{glossaries}{#1}%
286 }
```

Define package option to suppress warnings

```
287 \DeclareOptionX{nowarn}{%
288   \renewcommand*{\GlossariesWarning}[1]{}%
289   \renewcommand*{\GlossariesWarningNoLine}[1]{}%
290 }
```

compatible-2.07

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

Process package options:

```
293 \ProcessOptionsX
```

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

```
294 \ifglstranslate
295   \@ifpackageloaded{polyglossia}%
296   {%
```

polyglossia fakes babel so need to check for polyglossia first.

```
297   }%
298   {%
299     \@ifpackageloaded{babel}%
300     {%
301       \IfFileExists{translator.sty}%
302       {%
303         \RequirePackage{translator}%
304       }%
305     }%
306   }%
307   {}
308 }
309 \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}`).

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



```

311 {%
312   \ifcsundef{chapter}{}%
313   {%
314     \let\@gls@old@chapter\@chapter
315     \def\@chapter[#1]#2{\@gls@old@chapter[#1]{#2}%
316       \ifcsundef{hyperdef}{\hyperdef{section}{\thesection}}}%
317   }%
318 }%
319 {}

```

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

```

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

```

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

```

321 \newcommand*{\@onlypremakeg}[1]{%
322   \ifx\@gls@onlypremakeg\@empty
323     \def\@gls@onlypremakeg{#1}%
324   \else
325     \expandafter\toks@\expandafter{\@gls@onlypremakeg}%
326     \edef\@gls@onlypremakeg{\the\toks@,\noexpand#1}%
327   \fi}

```

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

```

328 \newcommand*{\@disable@onlypremakeg}{%
329   \@for\@thiscs:=\@gls@onlypremakeg\do{%
330     \expandafter\@disable@premakecs\@thiscs%
331   }}

```

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

```

332 \newcommand*{\@disable@premakecs}[1]{%
333   \def#1{\PackageError{glossaries}{\string#1\space may only be
334     used before \string\makeglossaries}{You can't use
335     \string#1\space after \string\makeglossaries}}%
336 }

```

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`

```

337 \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`

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

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

```
339 \newcommand*{\glsettoctitle}[1]{%
340 \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`

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

`\descriptionname`

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

`\symbolname`

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

`\pagelistname`

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

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

`glssymbolsgroupname`

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

`glnumbersgroupname`

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

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

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

`\seename`

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

`\andname`

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

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

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

```

350 \newcommand*{\addglossarytocaptions}[1]{%
351   \ifcsundef{captions#1}{}%
352   {%
353     \expandafter\let\expandafter\@gls@tmp\csname captions#1\endcsname
354     \expandafter\toks@\expandafter{\@gls@tmp
355     \renewcommand*{\glossaryname}{\translate{Glossary}}}%
356   }%
357   \expandafter\edef\csname captions#1\endcsname{\the\toks@}%
358 }%
359 }

```

```

360 \ifglstranslate

```

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

```

361 \@ifpackageloaded{translator}{%
362   \usedictionary{glossaries-dictionary}%
363   \addglossarytocaptions{portuges}%
364   \addglossarytocaptions{portuguese}%
365   \addglossarytocaptions{brazil}%
366   \addglossarytocaptions{brazilian}%
367   \addglossarytocaptions{danish}%
368   \addglossarytocaptions{dutch}%
369   \addglossarytocaptions{afrikaans}%
370   \addglossarytocaptions{english}%
371   \addglossarytocaptions{UKenglish}%
372   \addglossarytocaptions{USenglish}%
373   \addglossarytocaptions{american}%
374   \addglossarytocaptions{australian}%
375   \addglossarytocaptions{british}%
376   \addglossarytocaptions{canadian}%
377   \addglossarytocaptions{newzealand}%
378   \addglossarytocaptions{french}%
379   \addglossarytocaptions{frenchb}%
380   \addglossarytocaptions{francais}%
381   \addglossarytocaptions{acadian}%
382   \addglossarytocaptions{canadien}%
383   \addglossarytocaptions{german}%
384   \addglossarytocaptions{germanb}%
385   \addglossarytocaptions{austrian}%
386   \addglossarytocaptions{naustrian}%
387   \addglossarytocaptions{ngerman}%
388   \addglossarytocaptions{irish}%
389   \addglossarytocaptions{italian}%
390   \addglossarytocaptions{magyar}%
391   \addglossarytocaptions{hungarian}%
392   \addglossarytocaptions{polish}%
393   \addglossarytocaptions{spanish}%

```

```

394 \renewcommand*{\glsettoctitle}[1]{%
395 \ifthenelse{\equal{#1}{main}}{%
396 \translatelet{\glossarytoctitle}{Glossary}}{%
397 \ifthenelse{\equal{#1}{acronym}}{%
398 \translatelet{\glossarytoctitle}{Acronyms}}{%
399 \def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}}%
400 \renewcommand*{\glossaryname}{\translate{Glossary}}%
401 \renewcommand*{\acronymname}{\translate{Acronyms}}%
402 \renewcommand*{\entryname}{\translate{Notation (glossaries)}}%
403 \renewcommand*{\descriptionname}{%
404 \translate{Description (glossaries)}}%
405 \renewcommand*{\symbolname}{\translate{Symbol (glossaries)}}%
406 \renewcommand*{\pagelistname}{%
407 \translate{Page List (glossaries)}}%
408 \renewcommand*{\glssymbolsgroupname}{%
409 \translate{Symbols (glossaries)}}%
410 \renewcommand*{\glnumbersgroupname}{%
411 \translate{Numbers (glossaries)}}%
412 }{%
413 \@ifpackageloaded{polyglossia}%
414 {\RequirePackage{glossaries-polyglossia}}%
415 {%
416 \@ifpackageloaded{babel}{%
417 \RequirePackage{glossaries-babel}}}%
418 }}
419 \fi

```

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

```
420 \newcommand*{\glspostdescription}{.}
```

`\nopostdesc` Provide a means to suppress description terminator for a given entry. (Useful for entries with no description.) Has no effect outside the glossaries.

```
421 \newcommand*{\nopostdesc}{}

```

`\@nopostdesc` Suppress next description terminator.

```

422 \newcommand*{\@nopostdesc}{%
423 \let\org@glspostdescription\glspostdescription
424 \def\glspostdescription{%
425 \let\glspostdescription\org@glspostdescription}%
426 }

```

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

```
427 \newcommand{\glspar}{\par}
```

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

```

428 \ifglxindy
429 \newcommand{\setStyleFile}[1]{%

```

```

430 \renewcommand{\istfilename}{#1.xdy}}
431 \else
432 \newcommand{\setStyleFile}[1]{%
433 \renewcommand{\istfilename}{#1.ist}}
434 \fi

```

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

```

435 \@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`

```

436 \ifglsxindy
437 \def\istfilename{\jobname.xdy}
438 \else
439 \def\istfilename{\jobname.ist}
440 \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`

```

441 \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`

```

442 \newcommand*\glscompositor{.}

```

`\glsSetCompositor` Sets the compositor.

```

443 \newcommand*\glsSetCompositor[1]{%
444 \renewcommand*\glscompositor{#1}}

```

Only use before `\makeglossaries`

```

445 \@onlypremakeg\glsSetCompositor

```

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

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

```

446 \newcommand*\@glsAlphacompositor{\glscompositor}

```

`\sSetAlphaCompositor` Sets the alpha compositor.

```

447 \ifglsxindy
448   \newcommand*\glsSetAlphaCompositor[1]{%
449     \renewcommand*\@glsAlphacompositor{#1}}
450 \else
451   \newcommand*\glsSetAlphaCompositor[1]{%
452     \glsnoxindywarning\glsSetAlphaCompositor}
453 \fi

```

Can only be used before `\makeglossaries`

```

454 \@onlypremakeg\glsSetAlphaCompositor

```

`\gls@suffiF` 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.

```

455 \newcommand*\gls@suffiF{}

```

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

```

456 \newcommand*\glsSetSuffiF[1]{%
457   \renewcommand*\gls@suffiF{#1}}

```

Only has an effect when used before `\makeglossaries`

```

458 \@onlypremakeg\glsSetSuffiF

```

`\gls@suffiFF` 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.

```

459 \newcommand*\gls@suffiFF{}

```

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

```

460 \newcommand*\glsSetSuffiFF[1]{%
461   \renewcommand*\gls@suffiFF{#1}%
462 }

```

`\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”.

```

463 \ifcsundef{hyperlink}%
464 {%
465   \newcommand*\glsnumberformat[1]{#1}%

```

```

466 }%
467 {%
468   \newcommand*{\glsnumberformat}[1]{\glshypernumber{#1}}%
469 }

```

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
470 \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
471 \newcommand{\delimR}{--}

```

The glossary preamble is given by `\glossarypreamble`. This will appear after the glossary sectioning command, and before the `theglossary` environment. It is designed to allow the user to add information pertaining to the glossary (e.g. "page numbers in italic indicate the primary definition") therefore `\glossarypreamble` shouldn't be affected by the glossary style. (So if you define your own glossary style, don't have it change `\glossarypreamble`.) The preamble is empty by default. If you have multiple glossaries, and you want a different preamble for each glossary, you will need to use `\printglossary` for each glossary type, instead of `\printglossaries`, and redefine `\glossarypreamble` before each `\printglossary`.

```

\glossarypreamble
472 \newcommand*{\glossarypreamble}{}

```

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

```

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

```

```

\glossarypostamble
473 \newcommand*{\glossarypostamble}{}

```

```

\glossarysection

```

The sectioning command that starts a glossary is given by `\glossarysection`. (This does not form part of the glossary style, and so should not be changed by

a glossary style.) If `\phantomsection` is defined, it uses `\p@glossarysection`, otherwise it uses `\@glossarysection`.

```

474 \newcommand*\glossarysection[2][\@gls@title]{%
475   \def\@gls@title{#2}%
476   \ifcsundef{phantomsection}%
477   {%
478     \@glossarysection{#1}{#2}%
479   }%
480   {%
481     \p@glossarysection{#1}{#2}%
482   }%
483   \glossarymark{\glossarytoctitle}%
484 }

```

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

```

485 \ifcsundef{glossarymark}%
486 {%
487   \ifglsucmark
488     \newcommand{\glossarymark}[1]{%
489       \@mkboth{\MakeUppercase{#1}}{\MakeUppercase{#1}}%
490     }
491   \else
492     \newcommand{\glossarymark}[1]{\@mkboth{#1}{#1}}
493   \fi
494 }%
495 {%
496   \GlossariesWarning{overriding \string\glossarymark}%
497   \@ifclassloaded{memoir}%
498   {
499     \ifglsucmark
500       \renewcommand{\glossarymark}[1]{%
501         \@mkboth{\MakeUppercase{#1}}{\MakeUppercase{#1}}%
502       }
503     \else
504       \renewcommand{\glossarymark}[1]{%
505         \markboth{\memUHead{#1}}{\memUHead{#1}}%
506       }
507     \fi
508   }
509   {
510     \ifglsucmark
511       \renewcommand{\glossarymark}[1]{%
512         \@mkboth{\MakeUppercase{#1}}{\MakeUppercase{#1}}%
513       }
514     \else
515       \renewcommand{\glossarymark}[1]{\@mkboth{#1}{#1}}
516     \fi
517   }

```


518 }

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`

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

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

`\@glossarysection`

```
521 \newcommand*\@glossarysection[2]{%
522 \ifx\@@glossarysecstar\@empty
523   \csname\@@glossarysec\endcsname{#2}%
524 \else
525   \csname\@@glossarysec\endcsname*{#2}%
526   \@gls@toc{#1}{\@glossarysec}%
527 \fi
528 \@@glossaryseclabel}
```

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

`\@pglossarysection`

```
529 \newcommand*\@pglossarysection[2]{%
530 \glsclearpage
531 \phantomsection
532 \ifx\@@glossarysecstar\@empty
533   \csname\@@glossarysec\endcsname{#2}%
534 \else
535   \@gls@toc{#1}{\@glossarysec}%
536   \csname\@@glossarysec\endcsname*{#2}%
537 \fi
538 \@@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.

```
539 \newcommand*\gls@doclearpage{%
540   \ifthenelse{\equal{\@@glossarysec}{chapter}}{%
541     {%
542       \ifcsundef{cleardoublepage}{\clearpage}{\cleardoublepage}%
543     }%
544   }%
```

```

543 }%
544 {}%
545 }

```

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

```

546 \newcommand*{\glsclearpage}{\gls@docclearpage}

```

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`

```

547 \newcommand*{\@gls@toc}[2]{%
548 \ifglstoc
549   \ifglslnumberline
550     \addcontentsline{toc}{#2}{\numberline{}}#1}%
551   \else
552     \addcontentsline{toc}{#2}{#1}%
553   \fi
554 \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

```

555 \newcommand*{\glsnoxindywarning}[1]{%
556   \GlossariesWarning{Not in xindy mode --- ignoring \string#1}%
557 }

```

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

```

558 \ifglsxindy
559   \edef\@xdyattributes{\string"default\string"}%
560 \fi

```

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

```

561 \ifglsxindy
562   \edef\@xdyattributelist{}%
563 \fi

```

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

```

564 \ifglsxindy
565   \def\@xdylocref{}
566 \fi

```

\@gls@ifinlist

```

567 \newcommand*{\@gls@ifinlist}[4]{%
568   \def\@do@ifinlist##1,#1,##2\end@do@ifinlist{%
569     \def\@gls@listsuffix{##2}%
570     \ifx\@gls@listsuffix\@empty
571       #4%
572     \else
573       #3%
574     \fi
575   }%
576   \@do@ifinlist,#2,#1,\end@do@ifinlist
577 }

```

\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.

```

578 \ifglxsindy
579   \newcommand*{\@xdycounters}{\@glscounter}
580   \newcommand*\GlsAddXdyCounters[1]{%
581     \@for\@gls@ctr:=#1\do{%
582       \edef\@do@addcounter{%
583         \noexpand\@gls@ifinlist{\@gls@ctr}{\@xdycounters}{}%
584         {%
585           \noexpand\edef\noexpand\@xdycounters{\@xdycounters,%
586             \noexpand\@gls@ctr}%
587         }%
588       }%
589       \@do@addcounter
590     }
591   }

```

Only has an effect before \writeist:

```

592   \@onlypremakeg\GlsAddXdyCounters
593 \else
594   \newcommand*\GlsAddXdyCounters[1]{%
595     \glsnxindywarning\GlsAddXdyAttribute
596   }
597 \fi

```

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

```

598 \newcommand*\@disabled@glssaddxdycounters{%
599   \PackageError{glossaries}{\string\GlsAddXdyCounters\space
600   can't be used after \string\GlsAddXdyAttribute}{Move all
601   occurrences of \string\GlsAddXdyCounters\space before the first
602   instance of \string\GlsAddXdyAttribute}%
603 }

```

`\GlsAddXdyAttribute` Adds an attribute.

```

604 \ifglxindy
    First define internal command that adds an attribute for a given counter (2nd
    argument is the counter):
605   \newcommand*\@glsaddxdyattribute[2]{%
    Add to xindy attribute list
606     \edef\@xdyattributes{\@xdyattributes ^^J \string"#1\string" ^^J
607       \string"#2#1\string"}%
    Add to xindy markup location.
608     \expandafter\toks@\expandafter{\@xdylocref}%
609     \edef\@xdylocref{\the\toks@ ^^J%
610       (markup-locref
611       :open \string"\string~n%
612         \expandafter\string\csname glsX#2X#1\endcsname
613         \string" ^^J
614       :close \string"\string" ^^J
615       :attr \string"#2#1\string")}%
    Define associated attribute command \glsX<counter>X<attribute>{\<Hprefix>}{\<n>}
616     \expandafter\gdef\csname glsX#2X#1\endcsname##1##2{%
617       \setentrycounter{##1}{#2}\csname #1\endcsname{##2}%
618     }%
619   }

    High-level command:
620   \newcommand*\GlsAddXdyAttribute[1]{%
    Add to comma-separated attribute list
621     \ifx\@xdyattributelist\@empty
622       \edef\@xdyattributelist{#1}%
623     \else
624       \edef\@xdyattributelist{\@xdyattributelist,#1}%
625     \fi

    Iterate through all specified counters and add counter-dependent attributes:
626     \@for\@this@counter:=\@xdycounters\do{%
627       \protected@edef\gls@do@addxdyattribute{%
628         \noexpand\@glsaddxdyattribute{#1}{\@this@counter}%
629       }
630       \gls@do@addxdyattribute
631     }%

    All occurrences of \GlsAddXdyCounters must be used before this command
632     \let\GlsAddXdyCounters\@disabled@glsaddxdycounters
633   }

    Only has an effect before \writeist:
634     \@onlypremakeg\GlsAddXdyAttribute
635   \else
636     \newcommand*\GlsAddXdyAttribute[1]{%

```

```

637 \glsnoxywarning\GlsAddXdyAttribute}
638 \fi

```

redefinedattributes Add known attributes for all defined counters

```

639 \ifglxindy
640 \newcommand*{\@gls@addpredefinedattributes}{%
641   \GlsAddXdyAttribute{glsnumberformat}
642   \GlsAddXdyAttribute{textrm}
643   \GlsAddXdyAttribute{textsf}
644   \GlsAddXdyAttribute{texttt}
645   \GlsAddXdyAttribute{textbf}
646   \GlsAddXdyAttribute{textmd}
647   \GlsAddXdyAttribute{textit}
648   \GlsAddXdyAttribute{textup}
649   \GlsAddXdyAttribute{textsl}
650   \GlsAddXdyAttribute{textsc}
651   \GlsAddXdyAttribute{emph}
652   \GlsAddXdyAttribute{glshypernumber}
653   \GlsAddXdyAttribute{hyperrrm}
654   \GlsAddXdyAttribute{hypersf}
655   \GlsAddXdyAttribute{hypertt}
656   \GlsAddXdyAttribute{hyperbf}
657   \GlsAddXdyAttribute{hypermd}
658   \GlsAddXdyAttribute{hyperit}
659   \GlsAddXdyAttribute{hyperup}
660   \GlsAddXdyAttribute{hypersl}
661   \GlsAddXdyAttribute{hypersc}
662   \GlsAddXdyAttribute{hyperemph}
663 }
664 \else
665   \let\@gls@addpredefinedattributes\relax
666 \fi

```

\@xdyuseralphabets List of additional alphabets

```

667 \def\@xdyuseralphabets{}

```

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

```

668 \ifglxindy
669   \newcommand*{\GlsAddXdyAlphabet}[2]{%
670     \edef\@xdyuseralphabets{%
671       \@xdyuseralphabets ^^J
672       (define-alphabet "#1" (#2))}}
673 \else
674   \newcommand*{\GlsAddXdyAlphabet}[2]{%
675     \glsnoxywarning\GlsAddXdyAlphabet}
676 \fi

```

This code is only required for xindy:

677 \ifglxindy

ls@xdy@locationlist List of predefined location names.

```
678 \newcommand*{\@gls@xdy@locationlist}{%
679     roman-page-numbers,%
680     Roman-page-numbers,%
681     arabic-page-numbers,%
682     alpha-page-numbers,%
683     Alpha-page-numbers,%
684     Appendix-page-numbers,%
685     arabic-section-numbers%
686 }
```

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.

```
687 \protected@edef\@gls@roman{\@roman{0}\string"
688     \string"roman-numbers-lowercase\string" :sep \string"}}%
689 \@onelevel@sanitize\@gls@roman
690 \edef\@tmp{\string" \string"roman-numbers-lowercase\string"
691     :sep \string"}%
692 \@onelevel@sanitize\@tmp
693 \ifx\@tmp\@gls@roman
694     \expandafter
695     \edef\csname @gls@xdy@Lclass@roman-page-numbers\endcsname{%
696         \string"roman-numbers-lowercase\string"%
697     }%
698 \else
699     \expandafter
700     \edef\csname @gls@xdy@Lclass@roman-page-numbers\endcsname{
701         :sep \string"\@gls@roman\string"%
702     }%
703 \fi
```

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

```
704 \expandafter\def\csname @gls@xdy@Lclass@Roman-page-numbers\endcsname{%
705     \string"roman-numbers-uppercase\string"%
706 }
```

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

```
707 \expandafter\def\csname @gls@xdy@Lclass@arabic-page-numbers\endcsname{%
708     \string"arabic-numbers\string"%
709 }
```

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

```

710 \expandafter\def\csname @gls@xdy@Lclass@alpha-page-numbers\endcsname{%
711 \string"alpha\string"%
712 }%

@Alpha-page-numbers Upper case alphabetical (A, B, ...).
713 \expandafter\def\csname @gls@xdy@Lclass@Alpha-page-numbers\endcsname{%
714 \string"ALPHA\string"%
715 }%

appendix-page-numbers Appendix style locations (e.g. A-1, A-2, ..., B-1, B-2, ...). The separator is given
by \@glsAlphacompositor.
716 \expandafter\def\csname @gls@xdy@Lclass@Appendix-page-numbers\endcsname{%
717 \string"ALPHA\string"
718 :sep \string"\@glsAlphacompositor\string"
719 \string"arabic-numbers\string"%
720 }

arabic-section-numbers Section number style locations (e.g. 1.1, 1.2, ...). The compositor is given by
\@glscompositor.
721 \expandafter\def\csname @gls@xdy@Lclass@arabic-section-numbers\endcsname{%
722 \string"arabic-numbers\string"
723 :sep \string"\@glscompositor\string"
724 \string"arabic-numbers\string"%
725 }%

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

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

End of xindy-only block:
728 \fi

\GlsAddXdyLocation \GlsAddXdyLocation[<prefix-loc>]{<name>}{<definition>} Define a new lo-
cation called <name>. The definition must use xindy syntax. (Note that this
doesn't check to see if the location is already defined. That is left to xindy to
complain about.)
729 \ifglsxindy
730 \newcommand*\GlsAddXdyLocation{[3] [] {%
731 \def\@gls@tmp{#1}%
732 \ifx\@gls@tmp\@empty
733 \edef\@xdyuserlocationdefs{%
734 \@xdyuserlocationdefs ^^J%
735 (define-location-class \string"#2\string"^^J\space\space
736 \space(:sep \string"{ }\glsoopenbrace\string" #3
737 :sep \string"\glsclosebrace\string"))

```

```

738     }%
739   \else
740     \edef\@xdyuserlocationdefs{%
741       \@xdyuserlocationdefs ^^J%
742       (define-location-class \string"#2\string"^^J\space\space
743       \space(:sep "\glssopenbrace"
744         #1
745         :sep "\glsclosebrace\glssopenbrace" #3
746         :sep "\glsclosebrace"))
747     }%
748   \fi
749   \edef\@xdyuserlocationnames{%
750     \@xdyuserlocationnames^^J\space\space\space
751     \string"#1\string"}%
752 }

```

Only has an effect before \writeist:

```

753 \@onlypremakeg\GlsAddXdyLocation
754 \else
755   \newcommand*\GlsAddXdyLocation}[2]{%
756     \glsnnoxindywarning\GlsAddXdyLocation}
757 \fi

```

ylocationclassorder Define location class order

```

758 \ifglxindy
759   \edef\@xdylocationclassorder{^^J\space\space\space
760     \string"roman-page-numbers\string"^^J\space\space\space
761     \string"arabic-page-numbers\string"^^J\space\space\space
762     \string"arabic-section-numbers\string"^^J\space\space\space
763     \string"alpha-page-numbers\string"^^J\space\space\space
764     \string"Roman-page-numbers\string"^^J\space\space\space
765     \string"Alpha-page-numbers\string"^^J\space\space\space
766     \string"Appendix-page-numbers\string"
767     \@xdyuserlocationnames^^J\space\space\space
768     \string"see\string"
769   }
770 \fi

```

Change the location order.

yLocationClassOrder

```

771 \ifglxindy
772   \newcommand*\GlsSetXdyLocationClassOrder[1]{%
773     \def\@xdylocationclassorder{#1}}
774 \else
775   \newcommand*\GlsSetXdyLocationClassOrder[1]{%
776     \glsnnoxindywarning\GlsSetXdyLocationClassOrder}
777 \fi

```

\@xdysortrules Define sort rules


```

778 \ifglxindy
779   \def\@xdysortrules{}
780 \fi

```

`\GlsAddSortRule` Add a sort rule

```

781 \ifglxindy
782   \newcommand*\GlsAddSortRule[2]{%
783     \expandafter\toks@\expandafter{\@xdysortrules}%
784     \protected@edef\@xdysortrules{\the\toks@ ^^J
785       (sort-rule \string"#1\string" \string"#2\string"))}%
786   }
787 \else
788   \newcommand*\GlsAddSortRule[2]{%
789     \glsnxindywarning\GlsAddSortRule}
790 \fi

```

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

```

791 \ifglxindy
792   \def\@xdyrequiredstyles{tex}
793 \fi

```

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

```

794 \ifglxindy
795   \newcommand*\GlsAddXdyStyle[1]{%
796     \edef\@xdyrequiredstyles{\@xdyrequiredstyles,#1}}%
797 \else
798   \newcommand*\GlsAddXdyStyle[1]{%
799     \glsnxindywarning\GlsAddXdyStyle}
800 \fi

```

`\GlsSetXdyStyles` Reset the list of required styles

```

801 \ifglxindy
802   \newcommand*\GlsSetXdyStyles[1]{%
803     \edef\@xdyrequiredstyles{#1}}
804 \else
805   \newcommand*\GlsSetXdyStyles[1]{%
806     \glsnxindywarning\GlsSetXdyStyles}
807 \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.

```

808 \ifglxindy
809   \@ifpackageloaded{babel}{%

```

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

```

810 \def\findrootlanguage{\begingroup
811   \escapechar=-1\relax
   normalize \language to category 12 chars
812   \edef\language{\%
813     \expandafter\string\csname\language\endcsname}%
   disable babel.sty useless commands
814   \def\NeedsTeXFormat##1[##2]{}%
815   \def\ProvidesPackage##1[##2]{}%
816   \let\LdfInit\relax
817   \def\languageattribute##1##2{%
   change the meaning of \DeclareOption
818   \def\DeclareOption##1##2{%
   at \DeclareOption* we end
819   \ifx##1*\expandafter\endinput\else
   else we build a string with the first argument
820   \edef\testlanguage{\expandafter\string\csname##1\endcsname}%
   if \testlanguage and \language are the same we execute the second
   argument
821   \ifx\testlanguage\language##2\fi
822   \fi}
   almost all options of babel are \input{<name>.ldf}
823   \def\input##1{\stripldf##1}%
   we put the root language name in \rootlanguage
824   \def\stripldf##1.ldf{\gdef\rootlanguage{##1}}%
   now input babel.sty, using the primitive \input
825   @@input babel.sty
826   \endgroup}%
827   }{%
   hasn't been loaded, so check if has been loaded
828   \@ifpackageloaded{ngerman}{%
829     \def\findrootlanguage{%
830       \def\rootlanguage{german}}%
831   }{%
   Neither babel nor ngerman have been loaded, so assume the root language is
   English
832   \def\findrootlanguage{%
833     \def\rootlanguage{english}}%
834   }%
835   }%
836 \fi

```

`\rootlanguage` Set default root language to English.

```
837 \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.

```
838 \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.

```
839 \ifglxindy
840   \newcommand*\GlsSetXdyLanguage[2][\glsdefaultttype]{%
841     \ifglossaryexists{#1}{%
842       \expandafter\def\csname @xdy@#1@language\endcsname{#2}%
843     }{%
844       \PackageError{glossaries}{Can't set language type for
845         glossary type '#1' --- no such glossary}{%
846         You have specified a glossary type that doesn't exist}}
847   \else
848     \newcommand*\GlsSetXdyLanguage[2][]{%
849       \glsnoxywarning\GlsSetXdyLanguage}
850 \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.

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

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

```
852 \ifglxindy
853   \newcommand*\GlsSetXdyCodePage[1]{%
854     \renewcommand*\@gls@codepage{#1}%
855   }
856 \else
857   \newcommand*\GlsSetXdyCodePage[1]{%
858     \glsnoxywarning\GlsSetXdyCodePage}
859 \fi
```

`\@xdylettergroups` Store letter group definitions.

```
860 \ifglxindy
861   \ifglx@xindy@glsnumbers
862     \def\@xdylettergroups{(define-letter-group
863       \string"glsnumbers\string"^^J\space\space\space
864       :prefixes (\string"0\string" \string"1\string"
```

```

865      \string"2\string" \string"3\string" \string"4\string"
866      \string"5\string" \string"6\string" \string"7\string"
867      \string"8\string" \string"9\string")^^J\space\space\space
868      :before \string"\@glsfirstletter\string"))}
869  \else
870    \def\@xdylettergroups{}
871  \fi
872 \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.

```

873 \newcommand*\GlsAddLetterGroup[2]{%
874   \expandafter\toks@\expandafter{\@xdylettergroups}%
875   \protected@edef\@xdylettergroups{\the\toks@^^J%
876   (define-letter-group \string"#1\string"^^J\space\space\space#2)}}%
877 }%

```

1.5 Loops and conditionals

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

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

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

```

878 \newcommand*\forallglossaries[3][\@glo@types]{%
879   \@for#2:=#1\do{\ifx#2\@empty\else#3\fi}%
880 }

```

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

```
\forglsentries[<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.

```

881 \newcommand*\forglsentries[3][\glsdefaulttype]{%
882   \edef\@glo@list{\csname glolist@#1\endcsname}%
883   \@for#2:=\@glo@list\do{\ifx#2\@empty\else#3\fi}%
884 }

```

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

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

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

```

885 \newcommand*\forallglsentries[3][\@glo@types]{%

```

```

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

```

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

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

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

```

888 \newcommand{\ifglossaryexists}[3]{%
889   \ifcsundef{glo@#1@out}{#3}{#2}%
890 }

```

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

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

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

```

891 \newcommand{\ifglsentryexists}[3]{%
892   \ifcsundef{glo@#1@name}{#3}{#2}%
893 }

```

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

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

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

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

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

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

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

```

895 \newcommand{\glsdoifexists}[2]{%
896   \ifglsentryexists{#1}{#2}{%
897     \PackageError{glossaries}{Glossary entry ‘#1’ has not been
898     defined}{You need to define a glossary entry before you
899     can use it.}}%
900 }

```

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

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

```

901 \newcommand{\glsdoifnoexists}[2]{%
902   \ifglsentryexists{#1}{%
903     \PackageError{glossaries}{Glossary entry ‘#1’ has already

```

```

904     been defined}}{#2}%
905 }

```

```

\ifglshaschildren \ifglshaschildren{<label>}{<true part>}{<false part>}
906 \newcommand{\ifglshaschildren}[3]{%
907   \glsdoifexists{#1}%
908   {%
909     \def\do@glshaschildren{#3}%
910     \expandafter\for@gl@sentries\expandafter[\csname glo@#1@type\endcsname]
911     {\glo@label}%
912     {%
913       \letcs\glo@parent{glo@\glo@label @parent}%
914       \ifthenelse{\equal{#1}{\glo@parent}}{%
915         {%
916           \def\do@glshaschildren{#2}%
917           \@endfortrue
918         }%
919       }%
920     }%
921     \do@glshaschildren
922   }%
923 }

```

```

\ifglshasparent \ifglshaschildren{<label>}{<true part>}{<false part>}
924 \newcommand{\ifglshasparent}[3]{%
925   \glsdoifexists{#1}%
926   {%
927     \ifcsempy{glo@#1@parent}{#3}{#2}%
928   }%
929 }

```

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
930 \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`

```

931 \newcommand*{\newglossary}[5][glg]{%
932 \ifglossaryexists{#2}{%
933   \PackageError{glossaries}{Glossary type ‘#2’ already exists}{%
934     You can’t define a new glossary called ‘#2’ because it already
935     exists}%
936 }{%

  Check if default has been set
937   \ifx\glsdefaulttype\relax
938     \gdef\glsdefaulttype{#2}%
939   \fi

  Add this to the list of glossary types:
940   \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.
941   \expandafter\gdef\csname glolist@#2\endcsname{,}%

  Store details of this new glossary type:
942   \expandafter\def\csname @glotype@#2in\endcsname{#3}%
943   \expandafter\def\csname @glotype@#2out\endcsname{#4}%
944   \expandafter\def\csname @glotype@#2title\endcsname{#5}%
945   \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.
946   \ifcsundef{gls@#2@display}%
947   {%
948     \expandafter\gdef\csname gls@#2@display\endcsname{\glsdisplay}%
949   }%
950   {}%
951   \ifcsundef{gls@#2@displayfirst}%
952   {%
953     \expandafter\gdef\csname gls@#2@displayfirst\endcsname{%
954       \glsdisplayfirst
955     }%
956   }%
957   {}%
```

Define sort counter if required:

```
958 \@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.)

```
959 \@ifnextchar[{\@gls@setcounter{#2}}%
960   {\@gls@setcounter{#2}[\glscounter]}}%
961 }
```

`\altnewglossary`

```
962 \newcommand*{\altnewglossary}[3]{%
963   \newglossary[#2-glg]{#1}{#2-gls}{#2-glo}{#3}%
964 }
```

Only define new glossaries in the preamble:

```
965 \@onlypreamble{\newglossary}
```

Only define new glossaries before `\makeglossaries`

```
966 \@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`

```
967 \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`

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

Add counter to xindy list, if not already added:

```
970   \ifglsxindy
971     \GlsAddXdyCounters{#2}%
972   \fi
973 }
```

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

`\@gls@getcounter`

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

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

```
976 \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.

```
977 \define@key{glossentry}{name}{%
978 \def\@glo@name{#1}%
979 }
```

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.)

```
980 \define@key{glossentry}{description}{%
981 \def\@glo@desc{#1}%
982 }
```

descriptionplural

```
983 \define@key{glossentry}{descriptionplural}{%
984 \def\@glo@descplural{#1}%
985 }
```

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>`.

```
986 \define@key{glossentry}{sort}{%
987 \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.

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

```

989 \def\@glo@text{#1}%
990 }

```

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.

```

991 \define@key{glossentry}{plural}{%
992 \def\@glo@plural{#1}%
993 }

```

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.

```

994 \define@key{glossentry}{first}{%
995 \def\@glo@first{#1}%
996 }

```

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.

```

997 \define@key{glossentry}{firstplural}{%
998 \def\@glo@firstplural{#1}%
999 }

```

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).

```

1000 \define@key{glossentry}{symbol}{%
1001 \def\@glo@symbol{#1}%
1002 }

```

symbolplural

```

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

```

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

```

1006 \define@key{glossentry}{type}{%
1007 \def\@glo@type{#1}}

```

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

```
1008 \define@key{glossentry}{counter}{%
1009   \ifcsundef{c@#1}%
1010   {%
1011     \PackageError{glossaries}%
1012     {There is no counter called ‘#1’}%
1013     {%
1014       The counter key should have the name of a valid counter
1015       as its value%
1016     }%
1017   }%
1018   {%
1019     \def\@glo@counter{#1}%
1020   }%
1021 }
```

see The see key specifies a list of cross-references

```
1022 \define@key{glossentry}{see}{%
1023   \def\@glo@see{#1}%
1024   \@glo@seeautonumberlist
1025 }
```

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

```
1026 \define@key{glossentry}{parent}{%
1027 \def\@glo@parent{#1}}
```

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

```
1028 \define@choicekey{glossentry}{nonumberlist}[\val\nr]{true,false}[true]{%
1029   \ifcase\nr\relax
1030     \def\@glo@prefix{\glsonextpages}%
1031   \else
1032     \def\@glo@prefix{\glsnextpages}%
1033   \fi
1034 }
```

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

user1

```
1035 \define@key{glossentry}{user1}{%
1036   \def\@glo@useri{#1}%
1037 }
```

user2

```
1038 \define@key{glossentry}{user2}{%
1039   \def\@glo@userii{#1}%
1040 }
```

user3

```
1041 \define@key{glossentry}{user3}{%
1042   \def\@glo@useriii{#1}%
1043 }
```

user4

```
1044 \define@key{glossentry}{user4}{%
1045   \def\@glo@useriv{#1}%
1046 }
```

user5

```
1047 \define@key{glossentry}{user5}{%
1048   \def\@glo@userv{#1}%
1049 }
```

user6

```
1050 \define@key{glossentry}{user6}{%
1051   \def\@glo@uservi{#1}%
1052 }
```

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.

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

shortplural This key is provided for use by \newacronym.

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

long This key is provided for use by \newacronym.

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

longplural This key is provided for use by \newacronym.

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

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

```
1065 \newcommand*\@glsnname{%
1066   \PackageError{glossaries}{name key required in
1067     \string\newglossaryentry\space for entry '@@glo@label'}{You
1068     haven't specified the entry name}}
```

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

```

1069 \newcommand*{\@glsdefaultplural}{\@glo@text\glspluralsuffix}

```

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

```

1070 \newcommand*{\@gls@missingnumberlist}[1]{%
1071   ??%
1072   \ifglssavenumberlist
1073     \GlossariesWarning{Missing number list for entry ‘#1’.
1074       Maybe makeglossaries + rerun required.}%
1075   \else
1076     \PackageError{glossaries}%
1077       {Package option ‘savenumberlist=true’ required.}%
1078     {%
1079       You must use the ‘savenumberlist’ package option
1080       to reference location lists.%
1081     }%
1082   \fi
1083 }

```

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

```

1084 \newcommand*{\@glsdefaultsort}{\@glo@name}

```

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

```

1085 \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.)

```

1086 \newrobustcmd{\newglossaryentry}[2]{%

```

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

```

1087   \glsdoifnoexists{#1}%
1088   {%

```

Store label

```

1089     \def\@glo@label{#1}%

```

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

```

1090     \let\@glo@name\@glsnname

```

```

1091     \def\@glo@desc{%
1092       \PackageError{glossaries}
1093       {%
1094         description key required in \string\newglossaryentry\space
1095         for entry ‘\@glo@label’%
1096       }%
1097     {%
1098       You haven’t specified the entry description%
1099     }%
1100   }%

```

```

1101 \def\@glo@descplural{\@glo@desc}%
1102 \def\@glo@type{\glsdefaulttype}%
1103 \def\@glo@symbol{\relax}%

1104 \def\@glo@symbolplural{\@glo@symbol}%
1105 \def\@glo@text{\@glo@name}%
1106 \let\@glo@plural\@glsdefaultplural

```

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

```

1107 \let\@glo@first\relax

1108 \let\@glo@firstplural\relax

```

Set the default sort:

```

1109 \let\@glo@sort\@glsdefaultsort

```

Set the default counter:

```

1110 \def\@glo@counter{\@gls@getcounter{\@glo@type}}%
1111 \def\@glo@see{}%

1112 \def\@glo@parent{}%

1113 \def\@glo@prefix{}%

1114 \def\@glo@useri{}%
1115 \def\@glo@userii{}%
1116 \def\@glo@useriii{}%
1117 \def\@glo@useriv{}%
1118 \def\@glo@userv{}%
1119 \def\@glo@uservi{}%

1120 \def\@glo@short{}%
1121 \def\@glo@shortpl{}%
1122 \def\@glo@long{}%
1123 \def\@glo@longpl{}%

```

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

```

1124 \@newglossaryentryprehook

```

Extract key-val information from third parameter:

```

1125 \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.

```

1126 \ifcsundef{glolist@\@glo@type}%
1127 {%
1128   \PackageError{glossaries}%
1129   {Glossary type '\@glo@type' has not been defined}%

```

```

1130      {You need to define a new glossary type, before making entries
1131      in it}%
1132    }%
1133    {%
1134      \protected@edef\@glo@list@\csname glo@list@\@glo@type\endcsname}%
1135      \expandafter\xdef\csname glo@list@\@glo@type\endcsname{\@glo@list@{#1},}%
1136    }%

    Initialise level to 0.
1137    \gls@level=0\relax

    Has this entry been assigned a parent?
1138    \ifx\@glo@parent\@empty

    Doesn't have a parent. Set \glo@<label>@parent to empty.
1139      \expandafter\gdef\csname glo@#1@parent\endcsname{}%
1140    \else

    Has a parent. Check to ensure this entry isn't its own parent.
1141      \ifthenelse{\equal{#1}{\@glo@parent}}{%
1142        {%
1143          \PackageError{glossaries}{Entry '#1' can't be its own parent}{}%
1144          \def\@glo@parent{}%
1145          \expandafter\gdef\csname glo@#1@parent\endcsname{}%
1146        }%
1147      }%

    Check the parent exists:
1148      \ifglsentryexists{\@glo@parent}%
1149      {%

    Parent exists. Set \glo@<label>@parent.
1150      \expandafter\xdef\csname glo@#1@parent\endcsname{\@glo@parent}%

    Determine level.
1151      \gls@level=\csname glo@\@glo@parent @level\endcsname\relax
1152      \advance\gls@level by 1\relax

    If name hasn't been specified, use same as the parent name
1153      \ifx\@glo@name\@glsnoname
1154        \expandafter\let\expandafter\@glo@name
1155        \csname glo@\@glo@parent @name\endcsname

    If name and plural haven't been specified, use same as the parent
1156      \ifx\@glo@plural\@glsdefaultplural
1157        \expandafter\let\expandafter\@glo@plural
1158        \csname glo@\@glo@parent @plural\endcsname
1159      \fi
1160    \fi
1161  }%
1162  {%

```

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

```

1163      \PackageError{glossaries}%
1164      {%
1165          Invalid parent '@glo@parent'
1166          for entry '#1' - parent doesn't exist%
1167      }%
1168      {%
1169          Parent entries must be defined before their children%
1170      }%
1171      \def@glo@parent{%
1172          \expandafter\gdef\csname glo@#1@parent\endcsname{%
1173      }%
1174      }%
1175      \fi

```

Set the level for this entry

```

1176      \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.)

```

1177      \ifx\relax@glo@firstplural
1178          \ifx\relax@glo@first
1179              \def@glo@firstplural{\@glo@plural}%
1180              \def@glo@first{\@glo@text}%
1181          \else
1182              \def@glo@firstplural{\@glo@first\glspluralsuffix}%
1183          \fi
1184      \else
1185          \ifx\relax@glo@first
1186              \def@glo@first{\@glo@text}%
1187          \fi
1188      \fi

```

Define commands associated with this entry:

```

1189      \expandafter
1190      \protected\xdef\csname glo@#1@text\endcsname{\@glo@text}%
1191      \expandafter
1192      \protected\xdef\csname glo@#1@plural\endcsname{\@glo@plural}%
1193      \expandafter
1194      \protected\xdef\csname glo@#1@first\endcsname{\@glo@first}%
1195      \expandafter
1196      \protected\xdef\csname glo@#1@firstpl\endcsname{\@glo@firstplural}%
1197      \expandafter
1198      \protected\xdef\csname glo@#1@type\endcsname{\@glo@type}%
1199      \expandafter
1200      \protected\xdef\csname glo@#1@counter\endcsname{\@glo@counter}%

```



```

1201 \expandafter
1202 \protected@xdef\csname glo@#1@useri\endcsname{\@glo@useri}%
1203 \expandafter
1204 \protected@xdef\csname glo@#1@userii\endcsname{\@glo@userii}%
1205 \expandafter
1206 \protected@xdef\csname glo@#1@useriii\endcsname{\@glo@useriii}%
1207 \expandafter
1208 \protected@xdef\csname glo@#1@useriv\endcsname{\@glo@useriv}%
1209 \expandafter
1210 \protected@xdef\csname glo@#1@userv\endcsname{\@glo@userv}%
1211 \expandafter
1212 \protected@xdef\csname glo@#1@uservi\endcsname{\@glo@uservi}%
1213 \expandafter
1214 \protected@xdef\csname glo@#1@short\endcsname{\@glo@short}%
1215 \expandafter
1216 \protected@xdef\csname glo@#1@shortpl\endcsname{\@glo@shortpl}%
1217 \expandafter
1218 \protected@xdef\csname glo@#1@long\endcsname{\@glo@long}%
1219 \expandafter
1220 \protected@xdef\csname glo@#1@longpl\endcsname{\@glo@longpl}%
1221 \@gls@sanitizename
1222 \expandafter\protected@xdef\csname glo@#1@name\endcsname{\@glo@name}%

```

Set default numberlist if not defined:

```

1223 \ifcsundef{glo@#1@numberlist}%
1224 {%
1225 \csxdef{glo@#1@numberlist}{\noexpand\@gls@missingnumberlist{\@glo@label}}%
1226 }%
1227 {}%

```

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.

```

1228 \def\@glo@@desc{\@glo@first}%
1229 \ifx\@glo@desc\@glo@@desc
1230 \let\@glo@desc\@glo@first
1231 \fi
1232 \@gls@sanitizedesc
1233 \expandafter\protected@xdef\csname glo@#1@desc\endcsname{\@glo@desc}%
1234 \expandafter\protected@xdef\csname glo@#1@descplural\endcsname{\@glo@descplural}%

```

Set the sort key for this entry:

```

1235 \@gls@defsort{\@glo@type}{#1}%

1236 \def\@glo@@symbol{\@glo@text}%
1237 \ifx\@glo@symbol\@glo@@symbol
1238 \let\@glo@symbol\@glo@text
1239 \fi
1240 \@gls@sanitizesymbol
1241 \expandafter\protected@xdef\csname glo@#1@symbol\endcsname{\@glo@symbol}%
1242 \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):

```

1243 \expandafter\gdef\csname glo@#1@flagfalse\endcsname{%
1244 \expandafter\global\expandafter
1245 \let\csname ifglo@#1@flag\endcsname\iffalse
1246 }%
1247 \expandafter\gdef\csname glo@#1@flagtrue\endcsname{%
1248 \expandafter\global\expandafter
1249 \let\csname ifglo@#1@flag\endcsname\iftrue
1250 }%
1251 \csname glo@#1@flagfalse\endcsname

```

Sort out any cross-referencing if required.

```

1252 \ifx\@glo@see\@empty
1253 \else
1254 \protected@edef\@do@glsee{%
1255 \noexpand\@gls@fixbraces\noexpand\@glo@list\@glo@see
1256 \noexpand\@nil
1257 \noexpand\expandafter\noexpand\@glsee\noexpand\@glo@list{#1}}%
1258 \@do@glsee
1259 \fi
1260 }%

```

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

```

1261 \do@glo@storeentry{#1}%

```

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

```

1262 \@newglossaryentryposthook
1263 }

```

glossaryentryprehook Allow extra information to be added to glossary entries:

```

1264 \newcommand*{\@newglossaryentryprehook}{}

```

glossaryentryposthook Allow extra information to be added to glossary entries:

```

1265 \newcommand*{\@newglossaryentryposthook}{}

```

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

```

1266 \newcommand*{\glsmoveentry}[2]{%
1267 \edef\glo@type{\csname glo@#1@type\endcsname}%
1268 \def\glo@list{,}%
1269 \for@gl@sentries[\glo@type]{\glo@label}%
1270 {%
1271 \ifthenelse{\equal{\glo@label}{#1}}{\eappto\glo@list{\glo@label,}}%
1272 }%
1273 \cslet{glolist@\glo@type}{\glo@list}%
1274 \csdef{glo@#1@type}{#2}%
1275 }

```

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

```
1276 \ifglxindy
1277   \newcommand*{\@glossaryentryfield}{\string\@glossaryentryfield}
1278 \else
1279   \newcommand*{\@glossaryentryfield}{\string\glossaryentryfield}
1280 \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.)

```
1281 \ifglxindy
1282   \newcommand*{\@glossarysubentryfield}{%
1283     \string\@glossarysubentryfield}
1284 \else
1285   \newcommand*{\@glossarysubentryfield}{%
1286     \string\glossarysubentryfield}
1287 \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.)

```
1288 \newcommand{\@glo@storeentry}[1]{%
  Get the sort string and escape any special characters
1289 \protected@edef\@glo@sort{\csname glo@#1@sort\endcsname}%
1290 \@gls@checkmkidxchars\@glo@sort
  Same again for the name string.
1291 \protected@edef\@glo@name{\csname glo@#1@name\endcsname}%
1292 \@gls@checkmkidxchars\@glo@name
  Add the font command. (The backslash needs to be escaped for xindy.)
1293 \ifglxindy
1294   \protected@edef\@glo@name{\string\@gls@namefont{\@glo@name}}%
1295 \else
1296   \protected@edef\@glo@name{\string\gls@namefont{\@glo@name}}%
1297 \fi
  Get the description string and escape any special characters
1298 \protected@edef\@glo@desc{\csname glo@#1@desc\endcsname}%
1299 \@gls@checkmkidxchars\@glo@desc
  Same again for the symbol
1300 \protected@edef\@glo@symbol{\csname glo@#1@symbol\endcsname}%
1301 \@gls@checkmkidxchars\@glo@symbol
  Escape any special characters in the prefix
1302 \@gls@checkmkidxchars\@glo@prefix
```

Get the parent, if one exists

```
1303 \edef\@glo@parent{\csname glo@#1@parent\endcsname}%
```

Write the information to the glossary file.

```
1304 \ifglxindy
```

Store using xindy syntax.

```
1305 \ifx\@glo@parent\@empty
```

Entry doesn't have a parent

```
1306 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
1307 (\string"\@glo@sort\string" %
1308 \string"\@glo@prefix\@glossaryentryfield{#1}\@glo@name
1309 }\@glo@desc}\@glo@symbol}\string") %
1310 }%
1311 \else
```

Entry has a parent

```
1312 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
1313 \csname glo@\@glo@parent @index\endcsname
1314 (\string"\@glo@sort\string" %
1315 \string"\@glo@prefix\@glossarysubentryfield%
1316 {\csname glo@#1@level\endcsname}{#1}\@glo@name
1317 }\@glo@desc}\@glo@symbol}\string") %
1318 }%
1319 \fi
1320 \else
```

Store using makeindex syntax.

```
1321 \ifx\@glo@parent\@empty
```

Sanitize \@glo@prefix

```
1322 \@onelevel@sanitize\@glo@prefix
```

Entry doesn't have a parent

```
1323 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
1324 \@glo@sort\@gls@actualchar\@glo@prefix
1325 \@glossaryentryfield{#1}\@glo@name}\@glo@desc
1326 }\@glo@symbol}%
1327 }%
1328 \else
```

Entry has a parent

```
1329 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
1330 \csname glo@\@glo@parent @index\endcsname\@gls@levelchar
1331 \@glo@sort\@gls@actualchar\@glo@prefix
1332 \@glossarysubentryfield
1333 {\csname glo@#1@level\endcsname}{#1}\@glo@name}\@glo@desc
1334 }\@glo@symbol}%
1335 }%
1336 \fi
1337 \fi
1338 }
```

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`

```
1339 \newcommand*{\glsreset}[1]{%
1340 \glsdoifexists{#1}{%
1341 \expandafter\global\csname glo@#1@flagfalse\endcsname}}
```

As above, but with only a local effect:

`\glslocalreset`

```
1342 \newcommand*{\glslocalreset}[1]{%
1343 \glsdoifexists{#1}{%
1344 \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`

```
1345 \newcommand*{\glsunset}[1]{%
1346 \glsdoifexists{#1}{%
1347 \expandafter\global\csname glo@#1@flagtrue\endcsname}}
```

As above, but with only a local effect:

`\glslocalunset`

```
1348 \newcommand*{\glslocalunset}[1]{%
1349 \glsdoifexists{#1}{%
1350 \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`

```
1351 \newcommand*{\glsresetall}[1][\@glo@types]{%
1352 \forallglsentries[1]{\@glsentry}{%
1353 \glsreset{\@glsentry}}}
```

As above, but with only a local effect:

`\glslocalresetall`

```
1354 \newcommand*{\glslocalresetall}[1][\@glo@types]{%
1355 \forallglsentries[1]{\@glsentry}{%
1356 \glslocalreset{\@glsentry}}}
```

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

`\glsunsetall`

```
1357 \newcommand*{\glsunsetall}[1][\@gls@types]{%
1358 \forallglsentries[#1]{\@glsentry}{%
1359 \glsunset{\@glsentry}}}
```

As above, but with only a local effect:

`\glslocalunsetall`

```
1360 \newcommand*{\glslocalunsetall}[1][\@gls@types]{%
1361 \forallglsentries[#1]{\@glsentry}{%
1362 \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⟩}`

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

`\loadglsentries`

```
1363 \newcommand*{\loadglsentries}[2][\@gls@default]{%
1364 \let\@gls@default\glsdefaulttype
1365 \def\glsdefaulttype{#1}\input{#2}%
1366 \let\glsdefaulttype\@gls@default}
```

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

```
1367 \@onlypreamble{\loadglsentries}
```

1.10 Using glossary entries in the text

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

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

`\glstextformat`

```
1368 \newcommand*{\glstextformat}[1]{#1}
```

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

`\glsdisplayfirst`

```
1369 \newcommand*{\glsdisplayfirst}[4]{#1#4}
```

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

`\glsdisplay`

```
1370 \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`

```
1371 \newcommand*{\defglsdisplay}[2][\glsdefaulttype]{%
```

```
1372 \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`

```
1373 \newcommand*{\defglsdisplayfirst}[2][\glsdefaulttype]{%
```

```
1374 \expandafter\def\csname gls@#1@displayfirst\endcsname##1##2##3##4{#2}}
```

1.10.1 Links to glossary entries

The links to glossary entries all have a first optional argument that can be used to change the format and counter of the associated entry number. Except for `\glslink`, the commands like `\gls` have a final optional argument that can be used to insert additional text in the link (this will usually be appended, but can be redefined using `\defglsdisplay` and `\defglsdisplayfirst`). It goes against the \TeX norm to have an optional argument after the mandatory arguments, but it makes more sense to write, say, `\gls{label}[s]` rather than, say, `\gls[append=s]{label}`. Since these control sequences are defined to include the final square bracket, spaces will be ignored after them. This is likely to lead to confusion as most users would not expect, say, `\gls{<label>}` to ignore following spaces, so `\new@ifnextchar` from the package is required.

The following keys can be used in the first optional argument. The counter key checks that the value is the name of a valid counter.

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

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.

```
1389 \define@key{glslink}{format}{%
1390 \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.

```
1391 \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*[\langle options \rangle]{\langle label \rangle}{\langle text \rangle}`

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

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

`\glslink`

```
1392 \newrobustcmd*{\glslink}{%
1393 \ifstar\@sgls@link\@gls@@link}
```

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

```
1394 \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.

```
1395 \newcommand*{\@gls@@link}[3] [] {%
1396   \ifglsentryexists{#2}%
1397   {%
1398     \@gls@link[#1]{#2}{#3}%
1399   }{%
1400     \PackageError{glossaries}{Glossary entry ‘#2’ has not been
1401     defined}{You need to define a glossary entry before you
1402     can use it.}%
```

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

```
1403   \glstextformat{#3}%
1404 }%
1405 }
```

`\@gls@link`

```
1406 \def\@gls@link[#1]#2#3{%
```

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

```
1407   \leavevmode
1408   \def\glslabel{#2}%
1409   \def\@glsnumberformat{\glsnumberformat}%
1410   \edef\@gls@counter{\csname glo@#2@counter\endcsname}%
1411   \KV@glslink@hypertrue
1412   \setkeys{glslink}{#1}%
```

Store the entry's counter in `\theglsentrycounter`

```
1413   \@gls@saveentrycounter
```

Define sort key if necessary:

```
1414   \@gls@setsort{#2}%
```

```

1415 \do@wrglossary{#2}%
1416 \ifKV@glslink@hyper
1417 \glslink{\glolinkprefix#2}{\glstextformat{#3}}%
1418 \else
1419 \glstextformat{#3}\relax
1420 \fi
1421 }

```

\glolinkprefix

```

1422 \newcommand*{\glolinkprefix}{glo:}

```

\glentrycounter Set default value of entry counter

```

1423 \def\glentrycounter{\glscounter}%

```

\glsaveentrycounter Need to check if using equation counter in align environment:

```

1424 \newcommand*{\@glsaveentrycounter}{%
1425 \def\@gl@Hcounter{}%

```

Are we using equation counter?

```

1426 \ifthenelse{\equal{\@gl@counter}{equation}}{%
1427 {

```

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

```

1428 \ifcsundef{xatlevel@}%
1429 {%
1430 \edef\theglentrycounter{\expandafter\noexpand
1431 \csname the\@gl@counter\endcsname}%
1432 }%
1433 {%
1434 \ifx\xatlevel@\@empty
1435 \edef\theglentrycounter{\expandafter\noexpand
1436 \csname the\@gl@counter\endcsname}%
1437 \else
1438 \savecounters@
1439 \advance\c@equation by 1\relax
1440 \edef\theglentrycounter{\csname the\@gl@counter\endcsname}%

```

Check if hyperref version of this counter

```

1441 \ifcsundef{theH\@gl@counter}%
1442 {%
1443 \def\@gl@Hcounter{\theglentrycounter}%
1444 }%
1445 {%
1446 \def\@gl@Hcounter{\csname theH\@gl@counter\endcsname}%
1447 }%
1448 \protected@edef\theHglentrycounter{\@gl@Hcounter}%
1449 \restorecounters@
1450 \fi
1451 }%

```

```

1452 }%
1453 {%

```

Not using equation counter so no special measures:

```

1454 \edef\theglsentrycounter{\expandafter\noexpand
1455 \csname the\@gls@counter\endcsname}%
1456 }%

```

Check if hyperref version of this counter

```

1457 \ifx\@gls@Hcounter\@empty
1458 \ifcsundef{theH\@gls@counter}%
1459 {%
1460 \def\theHglentrycounter{\theglsentrycounter}%
1461 }%
1462 {%
1463 \protected@edef\theHglentrycounter{\expandafter\noexpand
1464 \csname theH\@gls@counter\endcsname}%
1465 }%
1466 \fi
1467 }

```

`\@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.

```

1468 \def\@set@glo@numformat#1#2#3#4{%
1469 \expandafter\@glo@check@mkidxrangechar#3\@nil
1470 \protected@edef#1{%
1471 \@glo@prefix setentrycounter[#4]{#2}%
1472 \expandafter\string\csname\@glo@suffix\endcsname
1473 }%
1474 \@gls@checkmkidxchars#1%
1475 }

```

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.

```

1476 \def\@glo@check@mkidxrangechar#1#2\@nil{%
1477 \if#1(\relax
1478 \def\@glo@prefix{(%
1479 \if\relax#2\relax
1480 \def\@glo@suffix{glsnumberformat}%
1481 \else
1482 \def\@glo@suffix{#2}%
1483 \fi
1484 \else

```

```

1485 \if#1)\relax
1486 \def\@glo@prefix{}}%
1487 \if\relax#2\relax
1488 \def\@glo@suffix{glsnumberformat}%
1489 \else
1490 \def\@glo@suffix{#2}%
1491 \fi
1492 \else
1493 \def\@glo@prefix{}\def\@glo@suffix{#1#2}%
1494 \fi
1495 \fi}

```

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

```

1496 \newcommand*{\@gls@escbsdq}[1]{%
1497 \def\@gls@checkedmkidx{}}%
1498 \let\gls@xdystring=#1\relax
1499 \@onelevel@sanitize\gls@xdystring
1500 \edef\do@gls@xdycheckbackslash{%
1501 \noexpand\@gls@xdycheckbackslash\gls@xdystring\noexpand\@nil
1502 \@backslashchar\@backslashchar\noexpand\null}%
1503 \do@gls@xdycheckbackslash
1504 \expandafter\@gls@updatechecked\@gls@checkedmkidx{\gls@xdystring}%
1505 \def\@gls@checkedmkidx{}}%
1506 \expandafter\@gls@xdycheckquote\gls@xdystring\@nil""\null
1507 \expandafter\@gls@updatechecked\@gls@checkedmkidx{\gls@xdystring}%
1508 \let#1=\gls@xdystring
1509 }

```

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

gls@checkmkidxchars

```

1510 \newcommand{\@gls@checkmkidxchars}[1]{%
1511 \ifglxsindy
1512 \@gls@escbsdq{#1}%
1513 \else
1514 \def\@gls@checkedmkidx{}}%
1515 \expandafter\@gls@checkquote#1\@nil""\null
1516 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
1517 \def\@gls@checkedmkidx{}}%
1518 \expandafter\@gls@checkescquote#1\@nil\""\null
1519 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
1520 \def\@gls@checkedmkidx{}}%
1521 \expandafter\@gls@checkescactual#1\@nil\?\?\null
1522 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
1523 \def\@gls@checkedmkidx{}}%
1524 \expandafter\@gls@checkactual#1\@nil??\null
1525 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
1526 \def\@gls@checkedmkidx{}}%

```

```

1527 \expandafter\@gls@checkbar#1\@nil||\null
1528 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
1529 \def\@gls@checkedmkidx{}%
1530 \expandafter\@gls@checkeschar#1\@nil\\|\null
1531 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
1532 \def\@gls@checkedmkidx{}%
1533 \expandafter\@gls@checklevel#1\@nil!!\null
1534 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
1535 \fi
1536 }

```

Update the control sequence and strip trailing \@nil:

\@gls@updatechecked

```

1537 \def\@gls@updatechecked#1\@nil#2{\def#2{#1}}

```

\@gls@tmpb Define temporary token

```

1538 \newtoks\@gls@tmpb

```

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

```

1539 \def\@gls@checkquote#1"#2"#3\null{%
1540 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1541 \toks@={#1}%
1542 \ifx\null#2\null
1543 \ifx\null#3\null
1544 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1545 \def\@gls@checkquote{\relax}%
1546 \else
1547 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1548 \@gls@quotechar\@gls@quotechar\@gls@quotechar\@gls@quotechar}%
1549 \def\@gls@checkquote{\@gls@checkquote#3\null}%
1550 \fi
1551 \else
1552 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1553 \@gls@quotechar\@gls@quotechar}%
1554 \ifx\null#3\null
1555 \def\@gls@checkquote{\@gls@checkquote#2""\null}%
1556 \else
1557 \def\@gls@checkquote{\@gls@checkquote#2"#3\null}%
1558 \fi
1559 \fi
1560 \@gls@checkquote}

```

\@gls@checkescquote Do the same for \:

```

1561 \def\@gls@checkescquote#1\"#2\"#3\null{%
1562 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1563 \toks@={#1}%
1564 \ifx\null#2\null
1565 \ifx\null#3\null

```

```

1566 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1567 \def\@@gls@checkescquote{\relax}%
1568 \else
1569 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1570 \@gls@quotechar\string"\@gls@quotechar
1571 \@gls@quotechar\string"\@gls@quotechar}%
1572 \def\@@gls@checkescquote{\@gls@checkescquote#3\null}%
1573 \fi
1574 \else
1575 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1576 \@gls@quotechar\string"\@gls@quotechar}%
1577 \ifx\null#3\null
1578 \def\@@gls@checkescquote{\@gls@checkescquote#2\""\null}%
1579 \else
1580 \def\@@gls@checkescquote{\@gls@checkescquote#2\"#3\null}%
1581 \fi
1582 \fi
1583 \@@gls@checkescquote}

```

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

```

1584 \def\@gls@checkescactual#1\?#2\?#3\null{%
1585 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1586 \toks@={#1}%
1587 \ifx\null#2\null
1588 \ifx\null#3\null
1589 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1590 \def\@@gls@checkescactual{\relax}%
1591 \else
1592 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1593 \@gls@quotechar\string"\@gls@actualchar
1594 \@gls@quotechar\string"\@gls@actualchar}%
1595 \def\@@gls@checkescactual{\@gls@checkescactual#3\null}%
1596 \fi
1597 \else
1598 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1599 \@gls@quotechar\string"\@gls@actualchar}%
1600 \ifx\null#3\null
1601 \def\@@gls@checkescactual{\@gls@checkescactual#2\?\?\null}%
1602 \else
1603 \def\@@gls@checkescactual{\@gls@checkescactual#2\?#3\null}%
1604 \fi
1605 \fi
1606 \@gls@checkescactual}

```

`\@gls@checkescbar` Similarly for `\|`:

```

1607 \def\@gls@checkescbar#1\|#2\|#3\null{%
1608 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1609 \toks@={#1}%
1610 \ifx\null#2\null

```

```

1611 \ifx\null#3\null
1612 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1613 \def\@gls@checkescbar{\relax}%
1614 \else
1615 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1616 \@gls@quotechar\string"\@gls@encapchar
1617 \@gls@quotechar\string"\@gls@encapchar}%
1618 \def\@gls@checkescbar{\@gls@checkescbar#3\null}%
1619 \fi
1620 \else
1621 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1622 \@gls@quotechar\string"\@gls@encapchar}%
1623 \ifx\null#3\null
1624 \def\@gls@checkescbar{\@gls@checkescbar#2\\\null}%
1625 \else
1626 \def\@gls@checkescbar{\@gls@checkescbar#2\|#3\null}%
1627 \fi
1628 \fi
1629 \@gls@checkescbar}

```

\@gls@checkesclevel Similarly for \!:

```

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

```

\@gls@checkbar and for |:

```

1653 \def\@gls@checkbar#1|#2|#3\null{%
1654 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1655 \toks@={#1}%

```

```

1656 \ifx\null#2\null
1657 \ifx\null#3\null
1658 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1659 \def\@gls@checkbar{\relax}%
1660 \else
1661 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1662 \@gls@quotechar\@gls@encapchar\@gls@quotechar\@gls@encapchar}%
1663 \def\@gls@checkbar{\@gls@checkbar#3\null}%
1664 \fi
1665 \else
1666 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1667 \@gls@quotechar\@gls@encapchar}%
1668 \ifx\null#3\null
1669 \def\@gls@checkbar{\@gls@checkbar#2||\null}%
1670 \else
1671 \def\@gls@checkbar{\@gls@checkbar#2|#3\null}%
1672 \fi
1673 \fi
1674 \@gls@checkbar}

```

\@gls@checklevel and for !:

```

1675 \def\@gls@checklevel#1!#2!#3\null{%
1676 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1677 \toks@={#1}%
1678 \ifx\null#2\null
1679 \ifx\null#3\null
1680 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1681 \def\@gls@checklevel{\relax}%
1682 \else
1683 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1684 \@gls@quotechar\@gls@levelchar\@gls@quotechar\@gls@levelchar}%
1685 \def\@gls@checklevel{\@gls@checklevel#3\null}%
1686 \fi
1687 \else
1688 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1689 \@gls@quotechar\@gls@levelchar}%
1690 \ifx\null#3\null
1691 \def\@gls@checklevel{\@gls@checklevel#2!!\null}%
1692 \else
1693 \def\@gls@checklevel{\@gls@checklevel#2!#3\null}%
1694 \fi
1695 \fi
1696 \@gls@checklevel}

```

\@gls@checkactual and for ?:

```

1697 \def\@gls@checkactual#1?#2?#3\null{%
1698 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1699 \toks@={#1}%
1700 \ifx\null#2\null

```



```

1701 \ifx\null#3\null
1702 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1703 \def\@@gls@checkactual{\relax}%
1704 \else
1705 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1706 \@gls@quotechar\@gls@actualchar\@gls@quotechar\@gls@actualchar}%
1707 \def\@@gls@checkactual{\@gls@checkactual#3\null}%
1708 \fi
1709 \else
1710 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1711 \@gls@quotechar\@gls@actualchar}%
1712 \ifx\null#3\null
1713 \def\@@gls@checkactual{\@gls@checkactual#2??\null}%
1714 \else
1715 \def\@@gls@checkactual{\@gls@checkactual#2?#3\null}%
1716 \fi
1717 \fi
1718 \@@gls@checkactual}

```

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

```

1719 \def\@gls@xdycheckquote#1"#2"#3\null{%
1720 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1721 \toks@={#1}%
1722 \ifx\null#2\null
1723 \ifx\null#3\null
1724 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1725 \def\@@gls@xdycheckquote{\relax}%
1726 \else
1727 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1728 \string\}\string\}%
1729 \def\@@gls@xdycheckquote{\@gls@xdycheckquote#3\null}%
1730 \fi
1731 \else
1732 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1733 \string\}%
1734 \ifx\null#3\null
1735 \def\@@gls@xdycheckquote{\@gls@xdycheckquote#2"\null}%
1736 \else
1737 \def\@@gls@xdycheckquote{\@gls@xdycheckquote#2"#3\null}%
1738 \fi
1739 \fi
1740 \@@gls@xdycheckquote
1741 }

```

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

```

\@gls@xdycheckbackslash
1742 \edef\def@gls@xdycheckbackslash{%
1743 \noexpand\def\noexpand\@gls@xdycheckbackslash##1\@backslashchar
1744 ##2\@backslashchar##3\noexpand\null{%

```

```

1745 \noexpand\@gls@tmpb=\noexpand\expandafter
1746   {\noexpand\@gls@checkedmkidx}%
1747 \noexpand\toks@={##1}%
1748 \noexpand\ifx\noexpand\null##2\noexpand\null
1749 \noexpand\ifx\noexpand\null##3\noexpand\null
1750 \noexpand\edef\noexpand\@gls@checkedmkidx{%
1751   \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@}%
1752 \noexpand\def\noexpand\@gls@xdycheckbackslash{\relax}%
1753 \noexpand\else
1754 \noexpand\edef\noexpand\@gls@checkedmkidx{%
1755   \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
1756   \@backslashchar\@backslashchar\@backslashchar\@backslashchar}%
1757 \noexpand\def\noexpand\@gls@xdycheckbackslash{%
1758   \noexpand\@gls@xdycheckbackslash##3\noexpand\null}%
1759 \noexpand\fi
1760 \noexpand\else
1761 \noexpand\edef\noexpand\@gls@checkedmkidx{%
1762   \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
1763   \@backslashchar\@backslashchar}%
1764 \noexpand\ifx\noexpand\null##3\noexpand\null
1765 \noexpand\def\noexpand\@gls@xdycheckbackslash{%
1766   \noexpand\@gls@xdycheckbackslash##2\@backslashchar
1767   \@backslashchar\noexpand\null}%
1768 \noexpand\else
1769 \noexpand\def\noexpand\@gls@xdycheckbackslash{%
1770   \noexpand\@gls@xdycheckbackslash##2\@backslashchar
1771   ##3\noexpand\null}%
1772 \noexpand\fi
1773 \noexpand\fi
1774 \noexpand\@gls@xdycheckbackslash
1775 }%
1776 }

```

Now go ahead and define \@gls@xdycheckbackslash

```

1777 \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.

```

1778 \ifcsundef{hyperlink}%
1779 {%
1780   \gdef\@glslink#1#2{#2}%
1781 }%
1782 {%
1783   \gdef\@glslink#1#2{\hyperlink{#1}{#2}}%
1784 }

```

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

```

1785 \newlength\gls@tmplen
1786 \ifcsundef{hypertarget}%
1787 {%
1788   \gdef\@glstarget#1#2{#2}%
1789 }%
1790 {%
1791   \gdef\@glstarget#1#2{%
1792     \settoheight{\gls@tmplen}{#2}%
1793     \raisebox{\gls@tmplen}{\hypertarget{#1}{}}#2%
1794   }%
1795 }

```

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

`\glsdisablehyper`

```

1796 \newcommand{\glsdisablehyper}{%
1797 \renewcommand*\@glslink[2]{##2}%
1798 \renewcommand*\@glstarget[2]{##2}}

```

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

`\glsenablehyper`

```

1799 \newcommand{\glsenablehyper}{%
1800 \renewcommand*\@glslink[2]{\hyperlink{##1}{##2}}%
1801 \renewcommand*\@glstarget[2]{%
1802   \settoheight{\gls@tmplen}{##2}%
1803   \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`

```

1804 \newrobustcmd*{\gls}{\@ifstar\@sgls\@gls}

```

Define the starred form:

```

\@sgls
1805 \newcommand*{\@sgls}[1] [] {\@gls[hyper=false,#1]}

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

\@gls
1806 \newcommand*{\@gls}[2] [] {%
1807 \new@ifnextchar [\@gls@{#1}{#2}]{\@gls@{#1}{#2} []}}

\@gls@ Read in the final optional argument:
1808 \def\@gls@#1#2[#3]{%
1809 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

Save options in \@gls@link@opts and label in \@gls@link@label
1810 \def\@gls@link@opts{#1}%
1811 \def\@gls@link@label{#2}%

Determine what the link text should be (this is stored in \@glo@text)
1812 \ifglsused{#2}%
1813 {%
1814   \def\@glo@text{%
1815     \csname gls@\@glo@type @display\endcsname
1816     {\glsentrytext{#2}}{\glsentrydesc{#2}}{\glsentrysymbol{#2}}{#3}}%
1817 }%
1818 {%
1819   \def\@glo@text{%
1820     \csname gls@\@glo@type @displayfirst\endcsname
1821     {\glsentryfirst{#2}}{\glsentrydesc{#2}}{\glsentrysymbol{#2}}{#3}}%
1822 }%

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.
1823 \ifglsused{#2}{%
1824   \@gls@link[#1]{#2}{\@glo@text}%
1825 }{%
1826   \gls@checkisacronymlist\@glo@type
1827   \ifthenelse{(\boolean{@glsisacronymlist})\AND
1828     \boolean{glsacrfootnote}) \OR \NOT\boolean{glshyperfirst}}{%
1829     \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
1830   }{%
1831     \@gls@link[#1]{#2}{\@glo@text}%
1832   }%
1833 }%

Indicate that this entry has now been used
1834 \glsunset{#2}%
1835 }

```

`\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`

```
1836 \newrobustcmd*{\Gls}{\@ifstar\@sGls\@Gls}
```

Define the starred form:

```
1837 \newcommand*{\@sGls}[1][\@Gls[hyper=false,#1]]
```

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

```
1838 \newcommand*{\@Gls}[2][\@Gls@link@label]
```

```
1839 \new@ifnextchar[\@Gls@{#1}{#2}]{\@Gls@{#1}{#2}[]}
```

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

```
1840 \def\@Gls@#1#2[#3]{%
```

```
1841 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
1842 \def\@gls@link@opts{#1}%
```

```
1843 \def\@gls@link@label{#2}%
```

```
1844 \def\@gls@label{#2}%
```

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

```
1845 \ifglsused{#2}{%
```

```
1846 {%
```

```
1847 \protected@edef\@glo@text{%
```

```
1848 \csname gls@\@glo@type @display\endcsname
```

```
1849 {\glsentrytext{#2}}{\glsentrydesc{#2}}%
```

```
1850 {\glsentrysymbol{#2}}{#3}}%
```

```
1851 }%
```

```
1852 {%
```

```
1853 \protected@edef\@glo@text{%
```

```
1854 \csname gls@\@glo@type @displayfirst\endcsname
```

```
1855 {\glsentryfirst{#2}}{\glsentrydesc{#2}}%
```

```
1856 {\glsentrysymbol{#2}}{#3}}%
```

```
1857 }%
```

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.

```
1858 \ifglsused{#2}{%
```

```
1859 \@gls@link[#1]{#2}{%
```

```
1860 \expandafter\makefirstuc\expandafter{\@glo@text}}%
```

```
1861 }{%
```

```
1862 \gls@checkisacronymlist\@glo@type
```

```
1863 \ifthenelse{(\boolean{@glsisacronymlist})\AND
```

```
1864 \boolean{glsacrfootnote}) \OR \NOT\boolean{glshyperfirst}}{%
```

```

1865 \gls@link[#1,hyper=false]{#2}{%
1866 \expandafter\makefirstuc\expandafter{\@glo@text}}}%
1867 }{%
1868 \gls@link[#1]{#2}{%
1869 \expandafter\makefirstuc\expandafter{\@glo@text}}}%
1870 }%
1871 }%

```

Indicate that this entry has now been used

```

1872 \glsunset{#2}}%
1873 }

```

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

\GLS

```

1874 \newrobustcmd*{\GLS}{\@ifstar\@sGLS\@GLS}

```

Define the starred form:

```

1875 \newcommand*{\@sGLS}[1][\@GLS[hyper=false,#1]]{

```

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

```

1876 \newcommand*{\@GLS}[2][\@GLS@{#1}{#2}]{\@GLS@{#1}{#2}[]}
1877 \new@ifnextchar[\@GLS@{#1}{#2}]{\@GLS@{#1}{#2}[]}

```

\@GLS@ Read in the final optional argument:

```

1878 \def\@GLS@#1#2[#3]{%
1879 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

```

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

```

1880 \def\@gls@link@opts{#1}%
1881 \def\@gls@link@label{#2}%

```

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

```

1882 \ifglsused{#2}{\def\@glo@text{%
1883 \csname gls@\@glo@type @display\endcsname
1884 {\glsentrytext{#2}}{\glsentrydesc{#2}}{\glsentrysymbol{#2}}{#3}}}%
1885 \def\@glo@text{%
1886 \csname gls@\@glo@type @displayfirst\endcsname
1887 {\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 hyperfirst=false package option is used.

```

1888 \ifglsused{#2}{%
1889 \gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}}%
1890 }{%
1891 \gls@checkisacronymlist\@glo@type
1892 \ifthenelse{(\boolean{glsisacronymlist}\AND
1893 \boolean{glsacrfootnote})\OR \NOT\boolean{glshyperfirst}}{%
1894 \gls@link[#1,hyper=false]{#2}{\MakeUppercase{\@glo@text}}}%

```

```

1895 }{%
1896   \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
1897 }%
1898 }%

```

Indicate that this entry has now been used

```

1899 \glsunset{#2}}%
1900 }

```

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

`\glspl`

```

1901 \newrobustcmd*{\glspl}{\@ifstar\sglspl\glspl}

```

Define the starred form:

```

1902 \newcommand*{\sglspl}[1][\@glspl[hyper=false,#1]]{

```

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

```

1903 \newcommand*{\glspl}[2][\@glspl@{#1}{#2}]{\@glspl@{#1}{#2}[]}
1904 \new@ifnextchar{\@glspl@{#1}{#2}}{\@glspl@{#1}{#2}[]}

```

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

```

1905 \def\@glspl@#1#2[#3]{%
1906 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
    Save options in \@gls@link@opts and label in \@gls@link@label
1907 \def\@gls@link@opts{#1}%
1908 \def\@gls@link@label{#2}%

```

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

```

1909 \ifglsused{#2}%
1910 {%
1911   \def\@glo@text{%
1912     \csname gls@\@glo@type @display\endcsname
1913     {\glsentryplural{#2}}{\glsentrydescplural{#2}}%
1914     {\glsentrysymbolplural{#2}}{#3}}%
1915 }%
1916 {%
1917   \def\@glo@text{%
1918     \csname gls@\@glo@type @displayfirst\endcsname
1919     {\glsentryfirstplural{#2}}{\glsentrydescplural{#2}}%
1920     {\glsentrysymbolplural{#2}}{#3}}%
1921 }%

```

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.

```

1922 \ifglsused{#2}{%
1923   \@gls@link[#1]{#2}{\@glo@text}%
1924 }{%

```

```

1925 \gls@checkisacronymlist\@glo@type
1926 \ifthenelse{(\boolean{@glsisacronymlist}\AND
1927 \boolean{glsacrfootnote}) \OR \NOT\boolean{glshyperfirst}}{%
1928 \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
1929 }{%
1930 \@gls@link[#1]{#2}{\@glo@text}%
1931 }%
1932 }%

```

Indicate that this entry has now been used

```

1933 \glsunset{#2}}%
1934 }

```

`\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`

```

1935 \newrobustcmd*{\Glspl}{\@ifstar\@sGlspl\@Glspl}

```

Define the starred form:

```

1936 \newcommand*{\@sGlspl}[1][\@Glspl[hyper=false,#1]]{

```

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

```

1937 \newcommand*{\@Glspl}[2][\@Glspl@{#1}{#2}]{\@Glspl@{#1}{#2}[]}
1938 \new@ifnextchar{\@Glspl@{#1}{#2}}{\@Glspl@{#1}{#2}[]}

```

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

```

1939 \def\@Glspl@#1#2[#3]{%
1940 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%
    Save options in \@gls@link@opts and label in \@gls@link@label
1941 \def\@gls@link@opts{#1}%
1942 \def\@gls@link@label{#2}%
1943 \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`.

```

1944 \ifglsused{#2}%
1945 {%
1946 \protected@edef\@glo@text{%
1947 \csname gls@\@glo@type @display\endcsname
1948 {\glsentryplural{#2}}{\glsentrydescplural{#2}}%
1949 {\glsentrysymbolplural{#2}}{#3}}%
1950 }%
1951 {%
1952 \protected@edef\@glo@text{%
1953 \csname gls@\@glo@type @displayfirst\endcsname

```



```

1954      {\glsentryfirstplural{#2}}{\glsentrydescplural{#2}}}%
1955      {\glsentrysymbolplural{#2}}{#3}}}%
1956 }%

```

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.

```

1957 \ifglsused{#2}{%
1958   \@gls@link[#1]{#2}{%
1959     \expandafter\makefirstuc\expandafter{\@glo@text}}%
1960 }{%
1961   \gls@checkisacronymlist\@glo@type
1962   \ifthenelse{(\boolean{@glsisacronymlist}\AND
1963     \boolean{glsacrfootnote}) \OR \NOT\boolean{glshyperfirst}}{%
1964     \@gls@link[#1,hyper=false]{#2}{%
1965       \expandafter\makefirstuc\expandafter{\@glo@text}}%
1966   }{%
1967     \@gls@link[#1]{#2}{%
1968       \expandafter\makefirstuc\expandafter{\@glo@text}}%
1969   }%
1970 }%

```

Indicate that this entry has now been used

```

1971 \glsunset{#2}}%
1972 }

```

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

`\GLSp1`

```

1973 \newrobustcmd*{\GLSp1}{\@ifstar\@sGLSp1\@GLSp1}

```

Define the starred form:

```

1974 \newcommand*{\@sGLSp1}[1][\@GLSp1[hyper=false,#1]]

```

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

```

1975 \newcommand*{\@GLSp1}[2][\@GLSp1@{#1}{#2}]{\@GLSp1@{#1}{#2}[]}
1976 \new@ifnextchar[\@GLSp1@{#1}{#2}]{\@GLSp1@{#1}{#2}[]}

```

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

```

1977 \def\@GLSp1@#1#2[#3]{%
1978 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

```

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

```

1979 \def\@gls@link@opts{#1}%
1980 \def\@gls@link@label{#2}%

```

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

```

1981 \ifglsused{#2}{\def\@glo@text{%
1982 \csname gls@\@glo@type @display\endcsname

```

```

1983 {\glsentryplural{#2}}{\glsentrydescplural{#2}}{%
1984 \glsentrysymbolplural{#2}}{#3}}}%
1985 \def\@glo@text{%
1986 \csname gls@\@glo@type @displayfirst\endcsname
1987 {\glsentryfirstplural{#2}}{\glsentrydescplural{#2}}{%
1988 \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.

```

1989 \ifglsused{#2}{%
1990   \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
1991 }{%
1992   \gls@checkisacronymlist\@glo@type
1993   \ifthenelse{(\boolean{@glsisacronymlist}\AND
1994     \boolean{glsacrfootnote}) \OR \NOT\boolean{glshyperfirst}}{%
1995     \@gls@link[#1,hyper=false]{#2}{\MakeUppercase{\@glo@text}}%
1996   }{%
1997     \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
1998   }%
1999 }%

```

Indicate that this entry has now been used

```

2000 \glsunset{#2}}%
2001 }

```

`\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:

```

2002 \newrobustcmd*{\glsdisp}{\@ifstar\@sglsdisp\@glsdisp}

```

Define the starred form:

```

\@sgls
2003 \newcommand*{\@sglsdisp}[1][]{\@glsdisp[hyper=false,#1]}

```

Defined the un-starred form.

```

\@glsdisp
2004 \newcommand*{\@glsdisp}[3][]{%
2005   \glsdoifexists{#2}{%
2006     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2007   \def\@gls@link@opts{#1}%
2008   \def\@gls@link@label{#2}%

```

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

```

2009 \ifglsused{#2}%
2010 {%
2011 \def\@glo@text{%
2012 \csname gls@\@glo@type @display\endcsname
2013 {#3}{\glsentrydesc{#2}}{\glsentrysymbol{#2}}{}}%
2014 }%
2015 {%
2016 \def\@glo@text{%
2017 \csname gls@\@glo@type @displayfirst\endcsname
2018 {#3}{\glsentrydesc{#2}}{\glsentrysymbol{#2}}{}}%
2019 }%

```

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.

```

2020 \ifglsused{#2}%
2021 {%
2022 \@gls@link[#1]{#2}{\@glo@text}%
2023 }%
2024 {%
2025 \gls@checkisacronymlist\@glo@type
2026 \ifthenelse{(\boolean{glsisacronymlist}\AND
2027 \boolean{glsacrfootnote}) \OR \NOT\boolean{glshyperfirst}}{%
2028 {%
2029 \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
2030 }%
2031 {%
2032 \@gls@link[#1]{#2}{\@glo@text}%
2033 }%
2034 }%

```

Indicate that this entry has now been used

```

2035 \glsunset{#2}%
2036 }%
2037 }

```

\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

```

2038 \newrobustcmd*{\gls@text}{\@ifstar\@sgls@text\@gls@text}

```

Define the starred form:

```

2039 \newcommand*{\@sgls@text}[1][\@gls@text[hyper=false,#1]]

```

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

```

2040 \newcommand*{\@gls@text}[2][\@gls@text@{#1}{#2}[]]{%
2041 \new@ifnextchar[\@gls@text@{#1}{#2}]{\@gls@text@{#1}{#2}[]}}

```

Read in the final optional argument:

```
2042 \def\@glstext@#1#2[#3]{%
2043 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%
```

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

```
2044 \protected@edef\@glo@text{\glsentrytext{#2}}%
```

Call \@gls@link

```
2045 \@gls@link[#1]{#2}{\@glo@text#3}%
2046 }%
2047 }
```

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

\GLStext

```
2048 \newrobustcmd*{\GLStext}{\@ifstar\@sGLStext\@GLStext}
```

Define the starred form:

```
2049 \newcommand*{\@sGLStext}[1][]{\@GLStext[hyper=false,#1]}
```

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

```
2050 \newcommand*{\@GLStext}[2][]{%
2051 \new@ifnextchar[{\@GLStext@{#1}{#2}}{\@GLStext@{#1}{#2}[]}]}
```

Read in the final optional argument:

```
2052 \def\@GLStext@#1#2[#3]{%
2053 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%
```

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

```
2054 \protected@edef\@glo@text{\glsentrytext{#2}}%
```

Call \@gls@link

```
2055 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2056 }%
2057 }
```

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

\Glstext

```
2058 \newrobustcmd*{\Glstext}{\@ifstar\@sGlstext\@Glstext}
```

Define the starred form:

```
2059 \newcommand*{\@sGlstext}[1][]{\@Glstext[hyper=false,#1]}
```

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

```
2060 \newcommand*{\@Glstext}[2][]{%
2061 \new@ifnextchar[{\@Glstext@{#1}{#2}}{\@Glstext@{#1}{#2}[]}]}
```

Read in the final optional argument:

```
2062 \def\@Glstext@#1#2[#3]{%
2063 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%
```

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

```
2064 \protected@edef\@glo@text{\glsentrytext{#2}}%
```

Call \@gls@link

```
2065 \@gls@link[#1]{#2}{%
```

```
2066   \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
```

```
2067 }%
```

```
2068 }
```

\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

```
2069 \newrobustcmd*{\glsfirst}{\@ifstar\sglsfirst\@glsfirst}
```

Define the starred form:

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

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

```
2071 \newcommand*{\@glsfirst}[2] [] {%
```

```
2072 \new@ifnextchar[{\@glsfirst@{#1}{#2}}{\@glsfirst@{#1}{#2} []}]}
```

Read in the final optional argument:

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

```
2074 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2075 \protected@edef\@glo@text{\glsentryfirst{#2}}%
```

Call \@gls@link

```
2076 \@gls@link[#1]{#2}{\@glo@text#3}%
```

```
2077 }%
```

```
2078 }
```

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

\Glsfirst

```
2079 \newrobustcmd*{\Glsfirst}{\@ifstar\sglsfirst\@Glsfirst}
```

Define the starred form:

```
2080 \newcommand*{\@sglsfirst}[1] [] {\@Glsfirst[hyper=false,#1]}
```

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

```
2081 \newcommand*{\@Glsfirst}[2] [] {%
```

```
2082 \new@ifnextchar[{\@Glsfirst@{#1}{#2}}{\@Glsfirst@{#1}{#2} []}]}
```

Read in the final optional argument:

```
2083 \def\@Glsfirst@#1#2[#3] {%
```

```
2084 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2085 \protected@edef\@glo@text{\glsentryfirst{#2}}%
```

Call \@gls@link

```
2086 \@gls@link[#1]{#2}{%
2087   \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2088 }%
2089 }
```

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

\GLSfirst

```
2090 \newrobustcmd*{\GLSfirst}{\@ifstar\@sGLSfirst\@GLSfirst}
```

Define the starred form:

```
2091 \newcommand*{\@sGLSfirst}[1][]{\@GLSfirst[hyper=false,#1]}
```

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

```
2092 \newcommand*{\@GLSfirst}[2][]{%
2093 \new@ifnextchar[\@GLSfirst@{#1}{#2}}{\@GLSfirst@{#1}{#2}[]}}
```

Read in the final optional argument:

```
2094 \def\@GLSfirst@#1#2[#3]{%
2095 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2096 \protected@edef\@glo@text{\glsentryfirst{#2}}%
```

Call \@gls@link

```
2097 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2098 }%
2099 }
```

\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

```
2100 \newrobustcmd*{\glsplural}{\@ifstar\@sglsplural\@glsplural}
```

Define the starred form:

```
2101 \newcommand*{\@sglsplural}[1][]{\@glsplural[hyper=false,#1]}
```

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

```
2102 \newcommand*{\@glsplural}[2][]{%
2103 \new@ifnextchar[\@glsplural@{#1}{#2}}{\@glsplural@{#1}{#2}[]}}
```

Read in the final optional argument:

```
2104 \def\@glsplural@#1#2[#3]{%
2105 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2106 \protected@edef\@glo@text{\glsentryplural{#2}}%
```

Call \@gls@link
2107 \@gls@link[#1]{#2}{\@glo@text#3}%
2108 }%
2109 }

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

\Glsplural

2110 \newrobustcmd*{\Glsplural}{\@ifstar\@sGlsplural\@Glsplural}

Define the starred form:

2111 \newcommand*{\@sGlsplural}[1][\@Glsplural[hyper=false,#1]]

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

2112 \newcommand*{\@Glsplural}[2][\@%

2113 \new@ifnextchar[\@Glsplural@{#1}{#2}]{\@Glsplural@{#1}{#2}[]}}

Read in the final optional argument:

2114 \def\@Glsplural@#1#2[#3]{%

2115 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

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

2116 \protected@edef\@glo@text{\glsentryplural{#2}}%

Call \@gls@link

2117 \@gls@link[#1]{#2}{%

2118 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%

2119 }%

2120 }

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

\GLSplural

2121 \newrobustcmd*{\GLSplural}{\@ifstar\@sGLSplural\@GLSplural}

Define the starred form:

2122 \newcommand*{\@sGLSplural}[1][\@GLSplural[hyper=false,#1]]

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

2123 \newcommand*{\@GLSplural}[2][\@%

2124 \new@ifnextchar[\@GLSplural@{#1}{#2}]{\@GLSplural@{#1}{#2}[]}}

Read in the final optional argument:

2125 \def\@GLSplural@#1#2[#3]{%

2126 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

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

2127 \protected@edef\@glo@text{\glsentryplural{#2}}%

Call \@gls@link

```
2128 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%  
2129 }%  
2130 }
```

\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

```
2131 \newrobustcmd*{\glsfirstplural}{\@ifstar\sglsfirstplural\glsfirstplural}
```

Define the starred form:

```
2132 \newcommand*{\sglsfirstplural}[1][\@glsfirstplural[hyper=false,#1]]{
```

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

```
2133 \newcommand*{\@glsfirstplural}[2][\@%  
2134 \new@ifnextchar[\@glsfirstplural@{#1}{#2}]{\@glsfirstplural@{#1}{#2}[]}}
```

Read in the final optional argument:

```
2135 \def\@glsfirstplural@#1#2[#3]{%  
2136 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2137 \protected@edef\@glo@text{\glsentryfirstplural{#2}}%
```

Call \@gls@link

```
2138 \@gls@link[#1]{#2}{\@glo@text#3}%  
2139 }%  
2140 }
```

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

\Glsfirstplural

```
2141 \newrobustcmd*{\Glsfirstplural}{\@ifstar\sglsfirstplural\Glsfirstplural}
```

Define the starred form:

```
2142 \newcommand*{\sglsfirstplural}[1][\@Glsfirstplural[hyper=false,#1]]{
```

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

```
2143 \newcommand*{\@Glsfirstplural}[2][\@%  
2144 \new@ifnextchar[\@Glsfirstplural@{#1}{#2}]{\@Glsfirstplural@{#1}{#2}[]}}
```

Read in the final optional argument:

```
2145 \def\@Glsfirstplural@#1#2[#3]{%  
2146 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2147 \protected@edef\@glo@text{\glsentryfirstplural{#2}}%
```


Call \@gls@link

```

2148 \@gls@link[#1]{#2}{%
2149   \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2150 }%
2151 }

```

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

\GLSfirstplural

```

2152 \newrobustcmd*{\GLSfirstplural}{\@ifstar\@sGLSfirstplural\GLSfirstplural}

```

Define the starred form:

```

2153 \newcommand*{\@sGLSfirstplural}[1][\@GLSfirstplural[hyper=false,#1]]{

```

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

```

2154 \newcommand*{\@GLSfirstplural}[2][\@GLSfirstplural@#1]{%
2155 \new@ifnextchar[\@GLSfirstplural@#1]{#2}}{\@GLSfirstplural@#1}{#2}[]}}

```

Read in the final optional argument:

```

2156 \def\@GLSfirstplural@#1#2[#3]{%
2157 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

```

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

```

2158 \protected@edef\@glo@text{\glsentryfirstplural{#2}}%

```

Call \@gls@link

```

2159 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2160 }%
2161 }

```

\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

```

2162 \newrobustcmd*{\glsname}{\@ifstar\@sglsname\glsname}

```

Define the starred form:

```

2163 \newcommand*{\@sglsname}[1][\@glsname[hyper=false,#1]]{

```

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

```

2164 \newcommand*{\@glsname}[2][\@glsname@#1]{%
2165 \new@ifnextchar[\@glsname@#1]{#2}}{\@glsname@#1}{#2}[]}}

```

Read in the final optional argument:

```

2166 \def\@glsname@#1#2[#3]{%
2167 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

```

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

```

2168 \protected@edef\@glo@text{\glsentryname{#2}}%

```

Call \@gls@link

```

2169 \@gls@link[#1]{#2}{\@glo@text#3}%
2170 }%
2171 }

```

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

\Glsname

```

2172 \newrobustcmd*{\Glsname}{\@ifstar\@sGlsname\@Glsname}

```

Define the starred form:

```

2173 \newcommand*{\@sGlsname}[1] [] {\@Glsname[hyper=false,#1]}

```

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

```

2174 \newcommand*{\@Glsname}[2] [] {%
2175 \new@ifnextchar [{\@Glsname@{#1}{#2}}{\@Glsname@{#1}{#2} []}}

```

Read in the final optional argument:

```

2176 \def\@Glsname@#1#2[#3] {%
2177 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2178 \protected@edef\@glo@text{\glsentryname{#2}}%

```

Call \@gls@link

```

2179 \@gls@link[#1]{#2}{%
2180 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2181 }%
2182 }

```

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

\GLSname

```

2183 \newrobustcmd*{\GLSname}{\@ifstar\@sGLSname\@GLSname}

```

Define the starred form:

```

2184 \newcommand*{\@sGLSname}[1] [] {\@GLSname[hyper=false,#1]}

```

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

```

2185 \newcommand*{\@GLSname}[2] [] {%
2186 \new@ifnextchar [{\@GLSname@{#1}{#2}}{\@GLSname@{#1}{#2} []}}

```

Read in the final optional argument:

```

2187 \def\@GLSname@#1#2[#3] {%
2188 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2189 \protected@edef\@glo@text{\glsentryname{#2}}%

```

Call \@gls@link

```
2190 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%  
2191 }%  
2192 }
```

\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

```
2193 \newrobustcmd*{\glsdesc}{\@ifstar\sglsdesc\@glsdesc}
```

Define the starred form:

```
2194 \newcommand*{\@sglsdesc}[1][]{\@glsdesc[hyper=false,#1]}
```

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

```
2195 \newcommand*{\@glsdesc}[2][]{%  
2196 \new@ifnextchar[{\@glsdesc@{#1}{#2}}{\@glsdesc@{#1}{#2}[]}]}
```

Read in the final optional argument:

```
2197 \def\@glsdesc@#1#2[#3]{%  
2198 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2199 \protected@edef\@glo@text{\glsentrydesc{#2}}%
```

Call \@gls@link

```
2200 \@gls@link[#1]{#2}{\@glo@text#3}%  
2201 }%  
2202 }
```

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

\Glsdesc

```
2203 \newrobustcmd*{\Glsdesc}{\@ifstar\sglsdesc\@Glsdesc}
```

Define the starred form:

```
2204 \newcommand*{\@sglsdesc}[1][]{\@Glsdesc[hyper=false,#1]}
```

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

```
2205 \newcommand*{\@Glsdesc}[2][]{%  
2206 \new@ifnextchar[{\@Glsdesc@{#1}{#2}}{\@Glsdesc@{#1}{#2}[]}]}
```

Read in the final optional argument:

```
2207 \def\@Glsdesc@#1#2[#3]{%  
2208 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2209 \protected@edef\@glo@text{\glsentrydesc{#2}}%
```

Call \@gls@link
2210 \@gls@link[#1]{#2}{%
2211 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2212 }%
2213 }
\GLSdesc behaves like \glsdesc except that the link text is converted to uppercase.

\GLSdesc

2214 \newrobustcmd*{\GLSdesc}{\@ifstar\@sGLSdesc\@GLSdesc}
Define the starred form:
2215 \newcommand*{\@sGLSdesc}[1][\@GLSdesc[hyper=false,#1]]
Defined the un-starred form. Need to determine if there is a final optional argument
2216 \newcommand*{\@GLSdesc}[2][\@GLSdesc@{#1}{#2}]{\@GLSdesc@{#1}{#2}[]}
2217 \new@ifnextchar[\@GLSdesc@{#1}{#2}]{\@GLSdesc@{#1}{#2}[]}
Read in the final optional argument:
2218 \def\@GLSdesc@#1#2[#3]{%
2219 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
Determine what the link text should be (this is stored in \@glo@text)
2220 \protected@edef\@glo@text{\glsentrydesc{#2}}%
Call \@gls@link
2221 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2222 }%
2223 }
\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

2224 \newrobustcmd*{\glsdescplural}{\@ifstar\@sglsdescplural\@glsdescplural}
Define the starred form:
2225 \newcommand*{\@sglsdescplural}[1][\@glsdescplural[hyper=false,#1]]
Defined the un-starred form. Need to determine if there is a final optional argument
2226 \newcommand*{\@glsdescplural}[2][\@glsdescplural@{#1}{#2}]{\@glsdescplural@{#1}{#2}[]}
2227 \new@ifnextchar[\@glsdescplural@{#1}{#2}]{\@glsdescplural@{#1}{#2}[]}
Read in the final optional argument:
2228 \def\@glsdescplural@#1#2[#3]{%
2229 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
Determine what the link text should be (this is stored in \@glo@text)
2230 \protected@edef\@glo@text{\glsentrydescplural{#2}}%
Call \@gls@link

Call \@gls@link
2231 \@gls@link[#1]{#2}{\@glo@text#3}%
2232 }%
2233 }
\Glsdescplural behaves like \glsdescplural except that the first letter is converted to uppercase.

\Glsdescplural

2234 \newrobustcmd*{\Glsdescplural}{\@ifstar\@sGlsdescplural\@Glsdescplural}

Define the starred form:

2235 \newcommand*{\@sGlsdescplural}[1][\@Glsdescplural[hyper=false,#1]]

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

2236 \newcommand*{\@Glsdescplural}[2][\@Glsdescplural@{#1}{#2}[]]

2237 \new@ifnextchar[\@Glsdescplural@{#1}{#2}]{\@Glsdescplural@{#1}{#2}[]}]

Read in the final optional argument:

2238 \def\@Glsdescplural@#1#2[#3]{%

2239 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

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

2240 \protected@edef\@glo@text{\glsentrydescplural{#2}}%

Call \@gls@link

2241 \@gls@link[#1]{#2}{%

2242 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%

2243 }%

2244 }

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

\GLSdescplural

2245 \newrobustcmd*{\GLSdescplural}{\@ifstar\@sGLSdescplural\@GLSdescplural}

Define the starred form:

2246 \newcommand*{\@sGLSdescplural}[1][\@GLSdescplural[hyper=false,#1]]

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

2247 \newcommand*{\@GLSdescplural}[2][\@GLSdescplural@{#1}{#2}[]]

2248 \new@ifnextchar[\@GLSdescplural@{#1}{#2}]{\@GLSdescplural@{#1}{#2}[]}]

Read in the final optional argument:

2249 \def\@GLSdescplural@#1#2[#3]{%

2250 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

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

2251 \protected@edef\@glo@text{\glsentrydescplural{#2}}%

Call \@gls@link

```
2252 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%  
2253 }%  
2254 }
```

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

\glssymbol

```
2255 \newrobustcmd*{\glssymbol}{\@ifstar\@sglssymbol\@glssymbol}
```

Define the starred form:

```
2256 \newcommand*{\@sglssymbol}[1][]{\@glssymbol[hyper=false,#1]}
```

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

```
2257 \newcommand*{\@glssymbol}[2][]{%  
2258 \new@ifnextchar[{\@glssymbol@{#1}{#2}}{\@glssymbol@{#1}{#2}[]}]}
```

Read in the final optional argument:

```
2259 \def\@glssymbol@#1#2[#3]{%  
2260 \glsdoifexists{#2}{\edef\@glo@type{\glentrytype{#2}}%
```

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

```
2261 \protected@edef\@glo@text{\glentrysymbol{#2}}%
```

Call \@gls@link

```
2262 \@gls@link[#1]{#2}{\@glo@text#3}%  
2263 }%  
2264 }
```

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

\Glsymbol

```
2265 \newrobustcmd*{\Glsymbol}{\@ifstar\@sGlsymbol\@Glsymbol}
```

Define the starred form:

```
2266 \newcommand*{\@sGlsymbol}[1][]{\@Glsymbol[hyper=false,#1]}
```

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

```
2267 \newcommand*{\@Glsymbol}[2][]{%  
2268 \new@ifnextchar[{\@Glsymbol@{#1}{#2}}{\@Glsymbol@{#1}{#2}[]}]}
```

Read in the final optional argument:

```
2269 \def\@Glsymbol@#1#2[#3]{%  
2270 \glsdoifexists{#2}{\edef\@glo@type{\glentrytype{#2}}%
```

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

```
2271 \protected@edef\@glo@text{\glentrysymbol{#2}}%
```

Call \@gls@link

```

2272 \@gls@link[#1]{#2}{%
2273   \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2274 }%
2275 }

```

\GLSsymbol behaves like \glssymbol except that the link text is converted to uppercase.

\GLSsymbol

```

2276 \newrobustcmd*{\GLSsymbol}{\@ifstar\@sGLSsymbol\@GLSsymbol}

```

Define the starred form:

```

2277 \newcommand*{\@sGLSsymbol}[1][\@GLSsymbol[hyper=false,#1]]{

```

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

```

2278 \newcommand*{\@GLSsymbol}[2][\@GLSsymbol@{#1}{#2}[]]{%
2279 \new@ifnextchar[\@GLSsymbol@{#1}{#2}]{\@GLSsymbol@{#1}{#2}[]}}

```

Read in the final optional argument:

```

2280 \def\@GLSsymbol@#1#2[#3]{%
2281 \glsoifexists{#2}{\edef\@glo@type{\glstrytype{#2}}%

```

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

```

2282 \protected@edef\@glo@text{\glstrytype{#2}}%

```

Call \@gls@link

```

2283 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2284 }%
2285 }

```

\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

```

2286 \newrobustcmd*{\glssymbolplural}{\@ifstar\@sglssymbolplural\@glssymbolplural}

```

Define the starred form:

```

2287 \newcommand*{\@sglssymbolplural}[1][\@glssymbolplural[hyper=false,#1]]{

```

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

```

2288 \newcommand*{\@glssymbolplural}[2][\@glssymbolplural@{#1}{#2}[]]{%
2289 \new@ifnextchar[\@glssymbolplural@{#1}{#2}]{\@glssymbolplural@{#1}{#2}[]}}

```

Read in the final optional argument:

```

2290 \def\@glssymbolplural@#1#2[#3]{%
2291 \glsoifexists{#2}{\edef\@glo@type{\glstrytype{#2}}%

```

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

```

2292 \protected@edef\@glo@text{\glstrytype{#2}}%

```

Call \@gls@link

```
2293 \@gls@link[#1]{#2}{\@glo@text#3}%  
2294 }%  
2295 }
```

\Glsymbolplural behaves like \glssymbolplural except that the first letter is converted to uppercase.

\Glsymbolplural

```
2296 \newrobustcmd*{\Glsymbolplural}{\@ifstar\@sGlsymbolplural\@Glsymbolplural}
```

Define the starred form:

```
2297 \newcommand*{\@sGlsymbolplural}[1][\@Glsymbolplural[hyper=false,#1]]
```

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

```
2298 \newcommand*{\@Glsymbolplural}[2][\@Glsymbolplural@{#1}{#2}[]]{%  
2299 \new@ifnextchar[\@Glsymbolplural@{#1}{#2}]{\@Glsymbolplural@{#1}{#2}[]}}
```

Read in the final optional argument:

```
2300 \def\@Glsymbolplural@#1#2[#3]{%  
2301 \glsdoifexists{#2}{\edef\@glo@type{\glstrytype{#2}}}%
```

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

```
2302 \protected@edef\@glo@text{\glstrytype{#2}}%
```

Call \@gls@link

```
2303 \@gls@link[#1]{#2}{%  
2304 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%  
2305 }%  
2306 }
```

\GLSsymbolplural behaves like \glssymbolplural except that the link text is converted to uppercase.

\GLSsymbolplural

```
2307 \newrobustcmd*{\GLSsymbolplural}{\@ifstar\@sGLSsymbolplural\@GLSsymbolplural}
```

Define the starred form:

```
2308 \newcommand*{\@sGLSsymbolplural}[1][\@GLSsymbolplural[hyper=false,#1]]
```

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

```
2309 \newcommand*{\@GLSsymbolplural}[2][\@GLSsymbolplural@{#1}{#2}[]]{%  
2310 \new@ifnextchar[\@GLSsymbolplural@{#1}{#2}]{\@GLSsymbolplural@{#1}{#2}[]}}
```

Read in the final optional argument:

```
2311 \def\@GLSsymbolplural@#1#2[#3]{%  
2312 \glsdoifexists{#2}{\edef\@glo@type{\glstrytype{#2}}}%
```

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

```
2313 \protected@edef\@glo@text{\glstrytype{#2}}%
```


Call \@gls@link

```
2314 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2315 }%
2316 }
```

\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

```
2317 \newrobustcmd*{\glsuseri}{\@ifstar\sglsuseri\@glsuseri}
```

Define the starred form:

```
2318 \newcommand*{\@sglsuseri}[1][\@glsuseri[hyper=false,#1]]
```

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

```
2319 \newcommand*{\@glsuseri}[2][\@%
2320 \new@ifnextchar[\@glsuseri@{#1}{#2}]{\@glsuseri@{#1}{#2}[]}]
```

Read in the final optional argument:

```
2321 \def\@glsuseri@#1#2[#3]{%
2322 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2323 \protected@edef\@glo@text{\glsentryuseri{#2}}%
```

Call \@gls@link

```
2324 \@gls@link[#1]{#2}{\@glo@text#3}%
2325 }%
2326 }
```

\Glsuseri behaves like \glsuseri except that the first letter is converted to uppercase.

\Glsuseri

```
2327 \newrobustcmd*{\Glsuseri}{\@ifstar\sglsuseri\@Glsuseri}
```

Define the starred form:

```
2328 \newcommand*{\@sglsuseri}[1][\@Glsuseri[hyper=false,#1]]
```

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

```
2329 \newcommand*{\@Glsuseri}[2][\@%
2330 \new@ifnextchar[\@Glsuseri@{#1}{#2}]{\@Glsuseri@{#1}{#2}[]}]
```

Read in the final optional argument:

```
2331 \def\@Glsuseri@#1#2[#3]{%
2332 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2333 \protected@edef\@glo@text{\glsentryuseri{#2}}%
```

Call \@gls@link
2334 \@gls@link[#1]{#2}{%
2335 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2336 }%
2337 }

\GLSuseri behaves like \glsuseri except that the link text is converted to uppercase.

\GLSuseri

2338 \newrobustcmd*{\GLSuseri}{\@ifstar\@sGLSuseri\@GLSuseri}

Define the starred form:

2339 \newcommand*{\@sGLSuseri}[1][\@GLSuseri[hyper=false,#1]]

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

2340 \newcommand*{\@GLSuseri}[2][\@GLSuseri@{#1}{#2}[]]{%
2341 \new@ifnextchar[\@GLSuseri@{#1}{#2}[]]{\@GLSuseri@{#1}{#2}[]}}

Read in the final optional argument:

2342 \def\@GLSuseri@#1#2[#3]{%
2343 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

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

2344 \protected@edef\@glo@text{\glsentryuseri{#2}}%

Call \@gls@link

2345 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2346 }%
2347 }

\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

2348 \newrobustcmd*{\glsuserii}{\@ifstar\@sglsuserii\@glsuserii}

Define the starred form:

2349 \newcommand*{\@sglsuserii}[1][\@glsuserii[hyper=false,#1]]

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

2350 \newcommand*{\@glsuserii}[2][\@glsuserii@{#1}{#2}[]]{%
2351 \new@ifnextchar[\@glsuserii@{#1}{#2}[]]{\@glsuserii@{#1}{#2}[]}}

Read in the final optional argument:

2352 \def\@glsuserii@#1#2[#3]{%
2353 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

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

2354 \protected@edef\@glo@text{\glsentryuserii{#2}}%

Call \@gls@link
2355 \@gls@link[#1]{#2}{\@glo@text#3}%
2356 }%
2357 }

\Glsuserii behaves like \glsuserii except that the first letter is converted to uppercase.

\Glsuserii

2358 \newrobustcmd*{\Glsuserii}{\@ifstar\@sGlsuserii\@Glsuserii}

Define the starred form:

2359 \newcommand*{\@sGlsuserii}[1][\@Glsuserii[hyper=false,#1]]

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

2360 \newcommand*{\@Glsuserii}[2][\@%

2361 \new@ifnextchar[\@Glsuserii@{#1}{#2}]{\@Glsuserii@{#1}{#2}[]}}

Read in the final optional argument:

2362 \def\@Glsuserii@#1#2[#3]{%

2363 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

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

2364 \protected@edef\@glo@text{\glsentryuserii{#2}}%

Call \@gls@link

2365 \@gls@link[#1]{#2}{%

2366 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%

2367 }%

2368 }

\GLSuserii behaves like \glsuserii except that the link text is converted to uppercase.

\GLSuserii

2369 \newrobustcmd*{\GLSuserii}{\@ifstar\@sGLSuserii\@GLSuserii}

Define the starred form:

2370 \newcommand*{\@sGLSuserii}[1][\@GLSuserii[hyper=false,#1]]

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

2371 \newcommand*{\@GLSuserii}[2][\@%

2372 \new@ifnextchar[\@GLSuserii@{#1}{#2}]{\@GLSuserii@{#1}{#2}[]}}

Read in the final optional argument:

2373 \def\@GLSuserii@#1#2[#3]{%

2374 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

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

2375 \protected@edef\@glo@text{\glsentryuserii{#2}}%

Call \@gls@link
2376 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2377 }%
2378 }

\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

2379 \newrobustcmd*{\glsuseriii}{\@ifstar\@sglsuseriii\@glsuseriii}

Define the starred form:

2380 \newcommand*{\@sglsuseriii}[1][\@glsuseriii[hyper=false,#1]]

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

2381 \newcommand*{\@glsuseriii}[2][\@%

2382 \new@ifnextchar[\@glsuseriii@{#1}{#2}]{\@glsuseriii@{#1}{#2}[]}}

Read in the final optional argument:

2383 \def\@glsuseriii@#1#2[#3]{%

2384 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

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

2385 \protected@edef\@glo@text{\glsentryuseriii{#2}}%

Call \@gls@link

2386 \@gls@link[#1]{#2}{\@glo@text#3}}%

2387 }%

2388 }

\Glsuseriii behaves like \glsuseriii except that the first letter is converted to uppercase.

\Glsuseriii

2389 \newrobustcmd*{\Glsuseriii}{\@ifstar\@sGlsuseriii\@Glsuseriii}

Define the starred form:

2390 \newcommand*{\@sGlsuseriii}[1][\@Glsuseriii[hyper=false,#1]]

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

2391 \newcommand*{\@Glsuseriii}[2][\@%

2392 \new@ifnextchar[\@Glsuseriii@{#1}{#2}]{\@Glsuseriii@{#1}{#2}[]}}

Read in the final optional argument:

2393 \def\@Glsuseriii@#1#2[#3]{%

2394 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

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

2395 \protected@edef\@glo@text{\glsentryuseriii{#2}}%

Call \@gls@link

```

2396 \@gls@link[#1]{#2}{%
2397   \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2398 }%
2399 }

```

\GLSuseriii behaves like \glsuseriii except that the link text is converted to uppercase.

\GLSuseriii

```

2400 \newrobustcmd*{\GLSuseriii}{\@ifstar\@sGLSuseriii\@GLSuseriii}

```

Define the starred form:

```

2401 \newcommand*{\@sGLSuseriii}[1][\@GLSuseriii[hyper=false,#1]]{

```

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

```

2402 \newcommand*{\@GLSuseriii}[2][\@GLSuseriii[hyper=false,#1]]{
2403 \new@ifnextchar[\@GLSuseriii@{#1}{#2}}{\@GLSuseriii@{#1}{#2}[]}}

```

Read in the final optional argument:

```

2404 \def\@GLSuseriii@#1#2[#3]{%
2405 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

```

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

```

2406 \protected@edef\@glo@text{\glsentryuseriii{#2}}%

```

Call \@gls@link

```

2407 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2408 }%
2409 }

```

\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

```

2410 \newrobustcmd*{\glsuseriv}{\@ifstar\@sglsuseriv\@glsuseriv}

```

Define the starred form:

```

2411 \newcommand*{\@sglsuseriv}[1][\@glsuseriv[hyper=false,#1]]{

```

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

```

2412 \newcommand*{\@glsuseriv}[2][\@glsuseriv[hyper=false,#1]]{
2413 \new@ifnextchar[\@glsuseriv@{#1}{#2}}{\@glsuseriv@{#1}{#2}[]}}

```

Read in the final optional argument:

```

2414 \def\@glsuseriv@#1#2[#3]{%
2415 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

```

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

```

2416 \protected@edef\@glo@text{\glsentryuseriv{#2}}%

```

Call \@gls@link
2417 \@gls@link[#1]{#2}{\@glo@text#3}%
2418 }%
2419 }

\Glsuseriv behaves like \glsuseriv except that the first letter is converted to uppercase.

\Glsuseriv

2420 \newrobustcmd*{\Glsuseriv}{\@ifstar\@sGlsuseriv\@Glsuseriv}

Define the starred form:

2421 \newcommand*{\@sGlsuseriv}[1][]{\@Glsuseriv[hyper=false,#1]}

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

2422 \newcommand*{\@Glsuseriv}[2][]{%

2423 \new@ifnextchar[\@Glsuseriv@{#1}{#2}]{\@Glsuseriv@{#1}{#2}[]}}

Read in the final optional argument:

2424 \def\@Glsuseriv@#1#2[#3]{%

2425 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

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

2426 \protected@edef\@glo@text{\glsentryuseriv{#2}}%

Call \@gls@link

2427 \@gls@link[#1]{#2}{%

2428 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%

2429 }%

2430 }

\GLSuseriv behaves like \glsuseriv except that the link text is converted to uppercase.

\GLSuseriv

2431 \newrobustcmd*{\GLSuseriv}{\@ifstar\@sGLSuseriv\@GLSuseriv}

Define the starred form:

2432 \newcommand*{\@sGLSuseriv}[1][]{\@GLSuseriv[hyper=false,#1]}

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

2433 \newcommand*{\@GLSuseriv}[2][]{%

2434 \new@ifnextchar[\@GLSuseriv@{#1}{#2}]{\@GLSuseriv@{#1}{#2}[]}}

Read in the final optional argument:

2435 \def\@GLSuseriv@#1#2[#3]{%

2436 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

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

2437 \protected@edef\@glo@text{\glsentryuseriv{#2}}%

Call \@gls@link

```
2438 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2439 }%
2440 }
```

\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

```
2441 \newrobustcmd*{\glsuserv}{\@ifstar\sglsuserv\@glsuserv}
```

Define the starred form:

```
2442 \newcommand*{\@sglsuserv}[1][\@glsuserv[hyper=false,#1]]
```

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

```
2443 \newcommand*{\@glsuserv}[2][\@%
2444 \new@ifnextchar[\@glsuserv@{#1}{#2}]{\@glsuserv@{#1}{#2}[]}]
```

Read in the final optional argument:

```
2445 \def\@glsuserv@#1#2[#3]{%
2446 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2447 \protected@edef\@glo@text{\glsentryuser{#2}}%
```

Call \@gls@link

```
2448 \@gls@link[#1]{#2}{\@glo@text#3}%
2449 }%
2450 }
```

\Glsuserv behaves like \glsuserv except that the first letter is converted to uppercase.

\Glsuserv

```
2451 \newrobustcmd*{\Glsuserv}{\@ifstar\sglsuserv\@Glsuserv}
```

Define the starred form:

```
2452 \newcommand*{\@sglsuserv}[1][\@Glsuserv[hyper=false,#1]]
```

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

```
2453 \newcommand*{\@Glsuserv}[2][\@%
2454 \new@ifnextchar[\@Glsuserv@{#1}{#2}]{\@Glsuserv@{#1}{#2}[]}]
```

Read in the final optional argument:

```
2455 \def\@Glsuserv@#1#2[#3]{%
2456 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2457 \protected@edef\@glo@text{\glsentryuser{#2}}%
```

Call \@gls@link

```

2458 \@gls@link[#1]{#2}{%
2459   \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2460 }%
2461 }

```

\GLSuserv behaves like \glsuserv except that the link text is converted to uppercase.

\GLSuserv

```

2462 \newrobustcmd*{\GLSuserv}{\@ifstar\@sGLSuserv\@GLSuserv}

```

Define the starred form:

```

2463 \newcommand*{\@sGLSuserv}[1][\@GLSuserv[hyper=false,#1]]{

```

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

```

2464 \newcommand*{\@GLSuserv}[2][\@GLSuserv@{#1}{#2}]{%
2465 \new@ifnextchar[\@GLSuserv@{#1}{#2}]{\@GLSuserv@{#1}{#2}[]}}

```

Read in the final optional argument:

```

2466 \def\@GLSuserv@#1#2[#3]{%
2467 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2468 \protected@edef\@glo@text{\glsentryuserv{#2}}%

```

Call \@gls@link

```

2469 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2470 }%
2471 }

```

\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

```

2472 \newrobustcmd*{\glsuservi}{\@ifstar\@sglsuservi\@glsuservi}

```

Define the starred form:

```

2473 \newcommand*{\@sglsuservi}[1][\@glsuservi[hyper=false,#1]]{

```

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

```

2474 \newcommand*{\@glsuservi}[2][\@glsuservi@{#1}{#2}]{%
2475 \new@ifnextchar[\@glsuservi@{#1}{#2}]{\@glsuservi@{#1}{#2}[]}}

```

Read in the final optional argument:

```

2476 \def\@glsuservi@#1#2[#3]{%
2477 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2478 \protected@edef\@glo@text{\glsentryuservi{#2}}%

```


Call \@gls@link
2479 \@gls@link[#1]{#2}{\@glo@text#3}%
2480 }%
2481 }

\Glsuservi behaves like \glsuservi except that the first letter is converted to uppercase.

\Glsuservi

2482 \newrobustcmd*{\Glsuservi}{\@ifstar\@sGlsuservi\@Glsuservi}

Define the starred form:

2483 \newcommand*{\@sGlsuservi}[1][]{\@Glsuservi[hyper=false,#1]}

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

2484 \newcommand*{\@Glsuservi}[2][]{%

2485 \new@ifnextchar[{\@Glsuservi@{#1}{#2}}{\@Glsuservi@{#1}{#2}[]}]}

Read in the final optional argument:

2486 \def\@Glsuservi@#1#2[#3]{%

2487 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

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

2488 \protected@edef\@glo@text{\glsentryuservi{#2}}%

Call \@gls@link

2489 \@gls@link[#1]{#2}{%

2490 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%

2491 }%

2492 }

\GLSuservi behaves like \glsuservi except that the link text is converted to uppercase.

\GLSuservi

2493 \newrobustcmd*{\GLSuservi}{\@ifstar\@sGLSuservi\@GLSuservi}

Define the starred form:

2494 \newcommand*{\@sGLSuservi}[1][]{\@GLSuservi[hyper=false,#1]}

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

2495 \newcommand*{\@GLSuservi}[2][]{%

2496 \new@ifnextchar[{\@GLSuservi@{#1}{#2}}{\@GLSuservi@{#1}{#2}[]}]}

Read in the final optional argument:

2497 \def\@GLSuservi@#1#2[#3]{%

2498 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

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

2499 \protected@edef\@glo@text{\glsentryuservi{#2}}%

Call \@gls@link
2500 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2501 }%
2502 }

Now deal with acronym related keys. First the short form:

\acrshort

2503 \newrobustcmd*{\acrshort}{\@ifstar\s@acrshort\@ns@acrshort}

Define the starred form:

2504 \newcommand*{\s@acrshort}[2][]{%
2505 \new@ifnextchar[{\@acrshort{hyper=false,#1}{#2}}%
2506 {\@acrshort{hyper=false,#1}{#2}[]}%
2507 }

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

2508 \newcommand*{\ns@acrshort}[2][]{%
2509 \new@ifnextchar[{\@acrshort{#1}{#2}}{\@acrshort{#1}{#2}[]}%
2510 }

Read in the final optional argument:

2511 \def\@acrshort#1#2[#3]{%
2512 \glsdoifexists{#2}%
2513 {%
2514 \edef\@glo@type{\glsentrytype{#2}}%

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

2515 \protected@edef\@glo@text{\glsentryshort{#2}}%

Call \@gls@link

2516 \@gls@link[#1]{#2}{\acronymfont{\@glo@text}#3}%
2517 }%
2518 }

\Acrshort

2519 \newrobustcmd*{\Acrshort}{\@ifstar\s@Acrshort\@ns@Acrshort}

Define the starred form:

2520 \newcommand*{\s@Acrshort}[2][]{%
2521 \new@ifnextchar[{\@Acrshort{hyper=false,#1}{#2}}%
2522 {\@Acrshort{hyper=false,#1}{#2}[]}%
2523 }

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

2524 \newcommand*{\ns@Acrshort}[2][]{%
2525 \new@ifnextchar[{\@Acrshort{#1}{#2}}{\@Acrshort{#1}{#2}[]}%
2526 }

Read in the final optional argument:

```
2527 \def\@Acrshort#1#2[#3]{%
2528   \glsdoifexists{#2}%
2529   {%
2530     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2531   \protected@edef\@glo@text{\glsentryshort{#2}}%
```

Call \@gls@link

```
2532   \@gls@link[#1]{#2}%
2533   {%
2534     \acronymfont{\expandafter\makefirstuc\expandafter{\@glo@text}}#3%
2535   }%
2536 }%
2537 }
```

\ACRshort

```
2538 \newrobustcmd*{\ACRshort}{\@ifstar\s@ACRshort\ns@ACRshort}
```

Define the starred form:

```
2539 \newcommand*{\s@ACRshort}[2][{}]{%
2540   \new@ifnextchar[{\@ACRshort{hyper=false,#1}{#2}}%
2541   {\@ACRshort{hyper=false,#1}{#2}[]}%
2542 }
```

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

```
2543 \newcommand*{\ns@ACRshort}[2][{}]{%
2544   \new@ifnextchar[{\@ACRshort{#1}{#2}}{\@ACRshort{#1}{#2}[]}%
2545 }
```

Read in the final optional argument:

```
2546 \def\@ACRshort#1#2[#3]{%
2547   \glsdoifexists{#2}%
2548   {%
2549     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2550   \protected@edef\@glo@text{\glsentryshort{#2}}%
```

Call \@gls@link

```
2551   \@gls@link[#1]{#2}{\acronymfont{\MakeUppercase{\@glo@text#3}}}%
2552   }%
2553 }
```

Short plural:

\acrshortpl

```
2554 \newrobustcmd*{\acrshortpl}{\@ifstar\s@acrshortpl\ns@acrshortpl}
```

Define the starred form:

```
2555 \newcommand*{\s@acrshortpl}[2] [] {%
2556   \new@ifnextchar[{\@acrshortpl{hyper=false,#1}{#2}}%
2557     {\@acrshortpl{hyper=false,#1}{#2} []}%
2558 }
```

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

```
2559 \newcommand*{\ns@acrshortpl}[2] [] {%
2560   \new@ifnextchar[{\@acrshortpl{#1}{#2}}{\@acrshortpl{#1}{#2} []}%
2561 }
```

Read in the final optional argument:

```
2562 \def\@acrshortpl#1#2[#3] {%
2563   \glsdoifexists{#2}%
2564   {%
2565     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2566   \protected@edef\@glo@text{\glsentryshortpl{#2}}%
```

Call \@gls@link

```
2567   \@gls@link[#1]{#2}{\acronymfont{\@glo@text}#3}%
2568 }%
2569 }
```

\Acrshortpl

```
2570 \newrobustcmd*{\Acrshortpl}{\@ifstar\s@Acrshortpl\ns@Acrshortpl}
```

Define the starred form:

```
2571 \newcommand*{\s@Acrshortpl}[2] [] {%
2572   \new@ifnextchar[{\@Acrshortpl{hyper=false,#1}{#2}}%
2573     {\@Acrshortpl{hyper=false,#1}{#2} []}%
2574 }
```

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

```
2575 \newcommand*{\ns@Acrshortpl}[2] [] {%
2576   \new@ifnextchar[{\@Acrshortpl{#1}{#2}}{\@Acrshortpl{#1}{#2} []}%
2577 }
```

Read in the final optional argument:

```
2578 \def\@Acrshortpl#1#2[#3] {%
2579   \glsdoifexists{#2}%
2580   {%
2581     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2582   \protected@edef\@glo@text{\glsentryshortpl{#2}}%
```

Call \@gls@link

```

2583   \@gls@link[#1]{#2}%
2584   {%
2585     \acronymfont{\expandafter\makefirstuc\expandafter{\@glo@text}}#3%
2586   }%
2587 }%
2588 }

```

\ACRshortpl

```

2589 \newrobustcmd*{\ACRshortpl}{\@ifstar\s@ACRshortpl\ns@ACRshortpl}

```

Define the starred form:

```

2590 \newcommand*{\s@ACRshortpl}[2] [] {%
2591   \new@ifnextchar[{\@ACRshortpl{hyper=false,#1}{#2}}%
2592     {\@ACRshortpl{hyper=false,#1}{#2} []}%
2593 }

```

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

```

2594 \newcommand*{\ns@ACRshortpl}[2] [] {%
2595   \new@ifnextchar[{\@ACRshortpl{#1}{#2}}{\@ACRshortpl{#1}{#2} []}%
2596 }

```

Read in the final optional argument:

```

2597 \def\@ACRshortpl#1#2[#3] {%
2598   \glsdoifexists{#2}%
2599   {%
2600     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2601   \protected@edef\@glo@text{\glsentryshortpl{#2}}%

```

Call \@gls@link

```

2602   \@gls@link[#1]{#2}{\acronymfont{\MakeUppercase{\@glo@text#3}}}%
2603 }%
2604 }

```

\acrlong

```

2605 \newrobustcmd*{\acrlong}{\@ifstar\s@acrlong\ns@acrlong}

```

Define the starred form:

```

2606 \newcommand*{\s@acrlong}[2] [] {%
2607   \new@ifnextchar[{\@acrlong{hyper=false,#1}{#2}}%
2608     {\@acrlong{hyper=false,#1}{#2} []}%
2609 }

```

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

```

2610 \newcommand*{\ns@acrlong}[2] [] {%
2611   \new@ifnextchar[{\@acrlong{#1}{#2}}{\@acrlong{#1}{#2} []}%
2612 }

```

Read in the final optional argument:

```
2613 \def\@acrlong#1#2[#3]{%
2614   \glsdoifexists{#2}%
2615   {%
2616     \edef\@glo@type{\glstrytype{#2}}%
2617     \protected@edef\@glo@text{\glstrylong{#2}}%
2618     \Call\@gls@link
2619   }%
2620 }
```

\Acrlong

```
2621 \newrobustcmd*{\Acrlong}{\@ifstar\s@Acrlong\@ns@Acrlong}
```

Define the starred form:

```
2622 \newcommand*{\s@Acrlong}[2][ ]{%
2623   \new@ifnextchar[{\@Acrlong{hyper=false,#1}{#2}}%
2624   {\@Acrlong{hyper=false,#1}{#2}[]}%
2625 }
```

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

```
2626 \newcommand*{\ns@Acrlong}[2][ ]{%
2627   \new@ifnextchar[{\@Acrlong{#1}{#2}}{\@Acrlong{#1}{#2}[]}%
2628 }
```

Read in the final optional argument:

```
2629 \def\@Acrlong#1#2[#3]{%
2630   \glsdoifexists{#2}%
2631   {%
2632     \edef\@glo@type{\glstrytype{#2}}%
2633     \protected@edef\@glo@text{\glstrylong{#2}}%
2634     \Call\@gls@link
2635   }%
2636   \expandafter\makefirstuc\expandafter{\@glo@text}#3%
2637 }%
2638 }%
2639 }
```

\ACRlong

```
2640 \newrobustcmd*{\ACRlong}{\@ifstar\s@ACRlong\@ns@ACRlong}
```

Define the starred form:

```
2641 \newcommand*{\s@ACRlong}[2] [] {%
2642   \new@ifnextchar[{\@ACRlong{hyper=false,#1}{#2}}%
2643     {\@ACRlong{hyper=false,#1}{#2} []}%
2644 }
```

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

```
2645 \newcommand*{\ns@ACRlong}[2] [] {%
2646   \new@ifnextchar[{\@ACRlong{#1}{#2}}{\@ACRlong{#1}{#2} []}%
2647 }
```

Read in the final optional argument:

```
2648 \def\@ACRlong#1#2[#3] {%
2649   \glsdoifexists{#2}%
2650   {%
2651     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2652   \protected@edef\@glo@text{\glsentrylong{#2}}%
```

Call \@gls@link

```
2653   \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2654   }%
2655 }
```

Short plural:

\acrlongpl

```
2656 \newrobustcmd*{\acrlongpl}{\@ifstar\s@acrlongpl\ns@acrlongpl}
```

Define the starred form:

```
2657 \newcommand*{\s@acrlongpl}[2] [] {%
2658   \new@ifnextchar[{\@acrlongpl{hyper=false,#1}{#2}}%
2659     {\@acrlongpl{hyper=false,#1}{#2} []}%
2660 }
```

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

```
2661 \newcommand*{\ns@acrlongpl}[2] [] {%
2662   \new@ifnextchar[{\@acrlongpl{#1}{#2}}{\@acrlongpl{#1}{#2} []}%
2663 }
```

Read in the final optional argument:

```
2664 \def\@acrlongpl#1#2[#3] {%
2665   \glsdoifexists{#2}%
2666   {%
2667     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2668   \protected@edef\@glo@text{\glsentrylongpl{#2}}%
```

```

Call \@gls@link
2669   \@gls@link[#1]{#2}{\@glo@text#3}%
2670   }%
2671 }

```

\Acrlongpl

```
2672 \newrobustcmd*{\Acrlongpl}{\@ifstar\s@Acrlongpl\ns@Acrlongpl}
```

Define the starred form:

```

2673 \newcommand*{\s@Acrlongpl}[2] [] {%
2674   \new@ifnextchar[{\@Acrlongpl{hyper=false,#1}{#2}}%
2675   {\@Acrlongpl{hyper=false,#1}{#2} []}%
2676 }

```

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

```

2677 \newcommand*{\ns@Acrlongpl}[2] [] {%
2678   \new@ifnextchar[{\@Acrlongpl{#1}{#2}}{\@Acrlongpl{#1}{#2} []}%
2679 }

```

Read in the final optional argument:

```

2680 \def\@Acrlongpl#1#2[#3] {%
2681   \glsdoifexists{#2}%
2682   {%
2683     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```
2684   \protected@edef\@glo@text{\glsentrylongpl{#2}}%
```

Call \@gls@link

```

2685   \@gls@link[#1]{#2}%
2686   {%
2687     \expandafter\makefirstuc\expandafter{\@glo@text}#3%
2688   }%
2689   }%
2690 }

```

\ACRlongpl

```
2691 \newrobustcmd*{\ACRlongpl}{\@ifstar\s@ACRlongpl\ns@ACRlongpl}
```

Define the starred form:

```

2692 \newcommand*{\s@ACRlongpl}[2] [] {%
2693   \new@ifnextchar[{\@ACRlongpl{hyper=false,#1}{#2}}%
2694   {\@ACRlongpl{hyper=false,#1}{#2} []}%
2695 }

```

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

```

2696 \newcommand*{\ns@ACRlongpl}[2] [] {%
2697   \new@ifnextchar[{\@ACRlongpl{#1}{#2}}{\@ACRlongpl{#1}{#2} []}%
2698 }

```


Read in the final optional argument:

```

2699 \def\@ACRlongpl#1#2[#3]{%
2700   \glsdoifexists{#2}%
2701   {%
2702     \edef\@glo@type{\glsentrytype{#2}}%
2703     \protected@edef\@glo@text{\glsentrylongpl{#2}}%
2704     \@gls@link
2705   }%
2706 }

```

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
2707 \newcommand*{\glsentryname}[1]{\csname glo@#1@name\endcsname}

```

```

\Glsentryname
2708 \newcommand*{\Glsentryname}[1]{%
2709 \protected@edef\@glo@text{\csname glo@#1@name\endcsname}%
2710 \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
2711 \newcommand*{\glsentrydesc}[1]{\csname glo@#1@desc\endcsname}

```

```

\Glsentrydesc
2712 \newcommand*{\Glsentrydesc}[1]{%
2713 \protected@edef\@glo@text{\csname glo@#1@desc\endcsname}%
2714 \expandafter\makefirstuc\expandafter{\@glo@text}}

```

Plural form:

`\glentrydescplural`

```
2715 \newcommand*{\glentrydescplural}[1]{%
2716 \csname glo@#1@descplural\endcsname}
```

`\Glsentrydescplural`

```
2717 \newcommand*{\Glsentrydescplural}[1]{%
2718 \protected@edef\@glo@text{\csname glo@#1@descplural\endcsname}%
2719 \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:

`\glentrytext`

```
2720 \newcommand*{\glentrytext}[1]{\csname glo@#1@text\endcsname}
```

`\Glsentrytext`

```
2721 \newcommand*{\Glsentrytext}[1]{%
2722 \protected@edef\@glo@text{\csname glo@#1@text\endcsname}%
2723 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Get the plural form:

`\glentryplural`

```
2724 \newcommand*{\glentryplural}[1]{\csname glo@#1@plural\endcsname}
```

`\Glsentryplural`

```
2725 \newcommand*{\Glsentryplural}[1]{%
2726 \protected@edef\@glo@text{\csname glo@#1@plural\endcsname}%
2727 \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.

`\glentrysymbol`

```
2728 \newcommand*{\glentrysymbol}[1]{\csname glo@#1@symbol\endcsname}
```

`\Glsentrysymbol`

```
2729 \newcommand*{\Glsentrysymbol}[1]{%
2730 \protected@edef\@glo@text{\csname glo@#1@symbol\endcsname}%
2731 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Plural form:

`\glentrysymbolplural`

```
2732 \newcommand*{\glentrysymbolplural}[1]{%
2733 \csname glo@#1@symbolplural\endcsname}
```

lentrysymbolplural

```
2734 \newcommand*{\Glsentrysymbolplural}[1]{%
2735 \protected@edef\@glo@text{\csname glo@#1@symbolplural\endcsname}%
2736 \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

```
2737 \newcommand*{\glsentryfirst}[1]{\csname glo@#1@first\endcsname}
```

\Glsentryfirst

```
2738 \newcommand*{\Glsentryfirst}[1]{%
2739 \protected@edef\@glo@text{\csname glo@#1@first\endcsname}%
2740 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Get the plural form (as specified by the firstplural key when the entry was defined).

glsentryfirstplural

```
2741 \newcommand*{\glsentryfirstplural}[1]{%
2742 \csname glo@#1@firstpl\endcsname}
```

Glsentryfirstplural

```
2743 \newcommand*{\Glsentryfirstplural}[1]{%
2744 \protected@edef\@glo@text{\csname glo@#1@firstpl\endcsname}%
2745 \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

```
2746 \newcommand*{\glsentrytype}[1]{\csname glo@#1@type\endcsname}
```

Display the sort text used for this entry. Note that the sort key is sanitized, so unexpected results may occur if the sort key contained commands.

\glsentrysort

```
2747 \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.

```
2748 \newcommand*{\glsentryuseri}[1]{\csname glo@#1@useri\endcsname}
```

\Glsentryuseri

```
2749 \newcommand*{\Glsentryuseri}[1]{%
2750 \protected@edef\@glo@text{\csname glo@#1@useri\endcsname}%
2751 \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.

```
2752 \newcommand*{\glsentryuserii}[1]{\csname glo@#1@userii\endcsname}
```

`\Glsentryuserii`

```
2753 \newcommand*{\Glsentryuserii}[1]{%
2754 \protected@edef\@glo@text{\csname glo@#1@userii\endcsname}%
2755 \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.

```
2756 \newcommand*{\glsentryuseriii}[1]{\csname glo@#1@useriii\endcsname}
```

`\Glsentryuseriii`

```
2757 \newcommand*{\Glsentryuseriii}[1]{%
2758 \protected@edef\@glo@text{\csname glo@#1@useriii\endcsname}%
2759 \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.

```
2760 \newcommand*{\glsentryuseriv}[1]{\csname glo@#1@useriv\endcsname}
```

`\Glsentryuseriv`

```
2761 \newcommand*{\Glsentryuseriv}[1]{%
2762 \protected@edef\@glo@text{\csname glo@#1@useriv\endcsname}%
2763 \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.

```
2764 \newcommand*{\glsentryuserv}[1]{\csname glo@#1@userv\endcsname}
```

`\Glsentryuserv`

```
2765 \newcommand*{\Glsentryuserv}[1]{%
2766 \protected@edef\@glo@text{\csname glo@#1@userv\endcsname}%
2767 \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.

```
2768 \newcommand*{\glsentryuservi}[1]{\csname glo@#1@uservi\endcsname}
```

`\Glsentryuservi`

```
2769 \newcommand*{\Glsentryuservi}[1]{%
2770 \protected@edef\@glo@text{\csname glo@#1@uservi\endcsname}%
2771 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

`\glsentryshort` Get the short key (as specified by the short the entry was defined). The argument is the label associated with the entry.

```
2772 \newcommand*{\glsentryshort}[1]{\csname glo@#1@short\endcsname}
```

```

\Glsentryshort
2773 \newcommand*{\Glsentryshort}[1]{%
2774 \protected@edef\@glo@text{\csname glo@#1@short\endcsname}%
2775 \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.
2776 \newcommand*{\glsentryshortpl}[1]{\csname glo@#1@shortpl\endcsname}

\Glsentryshortpl
2777 \newcommand*{\Glsentryshortpl}[1]{%
2778 \protected@edef\@glo@text{\csname glo@#1@shortpl\endcsname}%
2779 \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.
2780 \newcommand*{\glsentrylong}[1]{\csname glo@#1@long\endcsname}

\Glsentrylong
2781 \newcommand*{\Glsentrylong}[1]{%
2782 \protected@edef\@glo@text{\csname glo@#1@long\endcsname}%
2783 \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.
2784 \newcommand*{\glsentrylongpl}[1]{\csname glo@#1@longpl\endcsname}

\Glsentrylongpl
2785 \newcommand*{\Glsentrylongpl}[1]{%
2786 \protected@edef\@glo@text{\csname glo@#1@longpl\endcsname}%
2787 \expandafter\makefirstuc\expandafter{\@glo@text}}

Short cut macros to access full form:

\glsentryfull
2788 \newcommand*{\glsentryfull}[1]{%
2789   \glsentrylong{#1}\space(\glsentryshort{#1})%
2790 }

\Glsentryfull
2791 \newcommand*{\Glsentryfull}[1]{%
2792   \Glsentrylong{#1}\space(\glsentryshort{#1})%
2793 }

\glsentryfullpl
2794 \newcommand*{\glsentryfullpl}[1]{%
2795   \glsentrylongpl{#1}\space(\glsentryshortpl{#1})%
2796 }

```



```

2841         \def\@gls@donext@def{%
2842             \def\@gls@numlist@lastsep{\glsnumlistlastsep}%
2843         }%
2844     }%
2845     \expandafter \glsnumlistparser \expandafter{\the@numberlist}%
2846     \let\@gls@numlist@sep\@gls@numlist@lastsep
2847     \@gls@thislist
2848 \egroup
2849 }%
2850 }
2851 }

```

`\glsnumlistsep`

```
2852 \newcommand*{\glsnumlistsep}{, }
```

`\glsnumlistlastsep`

```
2853 \newcommand*{\glsnumlistlastsep}{ \& }
```

`\glshyperlink` Provide a hyperlink to a glossary entry without adding information to the glossary file. The entry needs to be added using a command like `\glslink` or `\glsadd` to ensure that the target is defined. The first (optional) argument specifies the link text. The entry name is used by default. The second argument is the entry label.

```

2854 \newcommand*{\glshyperlink}[2][\glsentrytext{\@glo@label}]{%
2855 \def\@glo@label{#2}%
2856 \@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`:

```

2857 \define@key{glossadd}{counter}{\def\@gls@counter{#1}}
2858 \define@key{glossadd}{format}{\def\@glsnumberformat{#1}}

```

This key is only used by `\glsaddall`:

```
2859 \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`

```

2860 \newrobustcmd*{\glsadd}[2][]{%
2861 \glsdoifexists{#2}%
2862 {%

```

```

2863 \def\@glsnumberformat{glsnumberformat}%
2864 \edef\@gls@counter{\csname glo@#2@counter\endcsname}%
2865 \setkeys{glossadd}{#1}%

Store the entry's counter in \theglsentrycounter
2866 \@gls@saveentrycounter
2867 \@do@wrglossary{#2}%
2868 }%
2869 }

```

`\glsaddall` [*glossary list*]

Add all terms defined for the listed glossaries (without displaying any text). If types key is omitted, apply to all glossary types.

```

\glsaddall
2870 \newrobustcmd*{\glsaddall}[1] [] {%
2871 \edef\@glo@type{\@glo@types}%
2872 \setkeys{glossadd}{#1}%
2873 \forallglsentries[\@glo@type]{\@glo@entry}{%
2874 \glsadd[#1]{\@glo@entry}}%
2875 }

```

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 `glsymbols` and `glsnumbers`, the group titles can be translated (so that `\glsymbolsgroupname` replaces `glsymbols` and `\glsnumbersgroupname` replaces `glsnumbers`) using the command `\glsgetgrouptitle` which is defined in `.`. This is done to prevent any problem characters in `\glsymbolsgroupname` and `\glsnumbersgroupname` from breaking hyperlinks.

`\glsopenbrace` Define `\glsopenbrace` to make it easier to write an opening brace to a file.

```
2876 \edef\glsopenbrace{\expandafter\@gobble\string\{}
```

`\glsclosebrace` Define `\glsclosebrace` to make it easier to write an opening brace to a file.

```
2877 \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.

```
2878 \edef\glsquote#1{\string"#1\string"}
```

`\@glsfirstletter` Define the first letter to come after the digits 0,...,9. Only required for xindy.

```
2879 \ifglsxindy
2880   \newcommand*{\@glsfirstletter}{A}
2881 \fi
```

`\stLetterAfterDigits` Sets the first letter to come after the digits 0,...,9.

```
2882 \ifglsxindy
2883   \newcommand*{\GlsSetXdyFirstLetterAfterDigits}[1]{%
2884     \renewcommand*{\@glsfirstletter}{#1}}
2885 \else
2886   \newcommand*{\GlsSetXdyFirstLetterAfterDigits}[1]{%
2887     \glsnoxywarning\GlsSetXdyFirstLetterAfterDigits}
2888 \fi
```

`\@glsminrange` Define the minimum number of successive location references to merge into a range.

```
2889 \newcommand*{\@glsminrange}{2}
```

`\setXdyMinRangeLength` Set the minimum range length. The value must either be none or a positive integer. The glossaries package doesn't check if the argument is valid, that is left to xindy.

```
2890 \ifglsxindy
2891   \newcommand*{\GlsSetXdyMinRangeLength}[1]{%
2892     \renewcommand*{\@glsminrange}{#1}}
2893 \else
2894   \newcommand*{\GlsSetXdyMinRangeLength}[1]{%
2895     \glsnoxywarning\GlsSetXdyMinRangeLength}
2896 \fi
```

`\writeist`

```
2897 \ifglsxindy
  Code to use if xindy is required.
2898   \def\writeist{%
    Update attributes list
2899     \@gls@addpredefinedattributes
    Open the file.
2900     \openout\glswrite=\istfilename
    Write header comment at the start of the file
2901     \write\glswrite{;; xindy style file created by the glossaries
2902       package}%
2903     \write\glswrite{;; for document '\jobname' on
2904       \the\year-\the\month-\the\day}%
```

Specify the required styles

```

2905 \write\glswrite{^^J; required styles^^J}
2906 \@for\@xdystyle:=\@xdyrequiredstyles\do{%
2907     \ifx\@xdystyle\@empty
2908     \else
2909     \protected@write\glswrite{}\{(require
2910     \string"\@xdystyle.xdy\string")}%
2911     \fi
2912 }%

```

List the allowed attributes (possible values used by the format key)

```

2913 \write\glswrite{^^J%
2914     ; list of allowed attributes (number formats)^^J}%
2915 \write\glswrite{(define-attributes ((\@xdyattributes)))}%

```

Define any additional alphabets

```

2916 \write\glswrite{^^J; user defined alphabets^^J}%
2917 \write\glswrite{\@xdyuseralphabets}%

```

Define location classes.

```

2918 \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.

```

2919 \@for\@gls@classI:=\@gls@xdy@locationlist\do{%

```

Case were $\langle Hprefix \rangle$ is empty:

```

2920 \protected@write\glswrite{}\{(define-location-class
2921 \string"\@gls@classI\string"^^J\space\space\space
2922 (
2923     :sep "{}{"
2924     \csname @gls@xdy@Lclass@\@gls@classI\endcsname\space
2925     :sep "}"
2926 )
2927 ^^J\space\space\space
2928 :min-range-length \@glsminrange^^J%
2929 )
2930 }%

```

Nested iteration over all classes:

```

2931 {%
2932     \@for\@gls@classII:=\@gls@xdy@locationlist\do{%
2933         \protected@write\glswrite{}\{(define-location-class
2934         \string"\@gls@classII-\@gls@classI\string"
2935         ^^J\space\space\space
2936         (
2937             :sep "{"
2938             \csname @gls@xdy@Lclass@\@gls@classII\endcsname\space
2939             :sep "{}{"
2940             \csname @gls@xdy@Lclass@\@gls@classI\endcsname\space
2941             :sep "}"

```

```

2942         )
2943         ^^J\space\space\space
2944         :min-range-length \@glsminrange^^J%
2945     )
2946 }%
2947 }%
2948 }%
2949 }%

```

User defined location classes (needs checking for new location format).

```

2950 \write\glswrite{^^J; user defined location classes}%
2951 \write\glswrite{\@xdyuserlocationdefs}%

```

Cross-reference class. (The unverified option is used as the cross-references are supplied using the list of labels along with the optional argument for `\glsseeformat` which xindy won't recognise.)

```

2952 \write\glswrite{^^J; define cross-reference class^^J}%
2953 \write\glswrite{(define-crossref-class \string"see\string"
2954 :unverified )}%

```

Define how cross-references should be displayed. This adds an empty set of braces after the cross-referencing information allowing for the final argument of `\glsseeformat` which gets ignored. (When using `makeindex` this final argument contains the location information which is not required.)

```

2955 \write\glswrite{(markup-crossref-list
2956 :class \string"see\string"^^J\space\space\space
2957 :open \string"\string\glsseeformat\string"
2958 :close \string"{}\string")}%

```

List the order to sort the classes.

```

2959 \write\glswrite{^^J; define the order of the location classes}%
2960 \write\glswrite{(define-location-class-order
2961 (\@xdylocationclassorder))}%

```

Specify what to write to the start and end of the glossary file.

```

2962 \write\glswrite{^^J; define the glossary markup^^J}%

2963 \write\glswrite{(markup-index^^J\space\space\space
2964 :open \string"\string
2965 \glossarysection[\string\glossarytoctitle]{\string
2966 \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)

```

2967 \@for\@this@ctr:=\@xdycounters\do{%
2968     {%
2969         \@for\@this@attr:=\@xdyattributelist\do{%
2970             \protected@write\glswrite{}{\string\providecommand*%
2971                 \expandafter\string
2972                 \csname glsX\@this@ctr X\@this@attr\endcsname[2]%
2973                 {%

```

```

2974         \string\setentrycounter
2975         [\expandafter\@gobble\string\#1]{\@this@ctr}%
2976         \expandafter\string
2977         \csname\@this@attr\endcsname
2978         {\expandafter\@gobble\string\#2}%
2979     }%
2980 }%
2981 }%
2982 }%
2983 }%

```

Add the end part of the open tag and the rest of the markup-index information:

```

2984     \write\glswrite{%
2985         \string\begin
2986         {theglossary}\string\glossaryheader\string~n\string" ^^J\space
2987         \space\space:close \string"\expandafter\@gobble
2988         \string%\string~n\string
2989         \end{theglossary}\string\glossarypostamble
2990         \string~n\string" ^^J\space\space\space
2991         :tree)}}%

```

Specify what to put between letter groups

```

2992     \write\glswrite{(markup-letter-group-list
2993         :sep \string"\string\glsgroupskip\string~n\string"))}%

```

Specify what to put between entries

```

2994     \write\glswrite{(markup-indexentry
2995         :open \string"\string\relax \string\glsresetentrylist
2996         \string~n\string"))}%

```

Specify how to format entries

```

2997     \write\glswrite{(markup-locclass-list :open
2998         \string"\glsopenbrace\string\glossaryentrynumbers
2999         \glsopenbrace\string\relax\space \string"^^J\space\space\space
3000         :sep \string", \string"
3001         :close \string"\glsclosebrace\glsclosebrace\string"))}%

```

Specify how to separate location numbers

```

3002     \write\glswrite{(markup-locref-list
3003         :sep \string"\string\delimN\space\string"))}%

```

Specify how to indicate location ranges

```

3004     \write\glswrite{(markup-range
3005         :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.

```

3006     \@onelevel@sanitize\gls@suffixF
3007     \@onelevel@sanitize\gls@suffixFF
3008     \ifx\gls@suffixF\@empty
3009     \else

```

```

3010     \write\glswrite{(markup-range
3011         :close "\gls@suffixF" :length 1 :ignore-end)}}%
3012 \fi
3013 \ifx\gls@suffixFF\@empty
3014 \else
3015     \write\glswrite{(markup-range
3016         :close "\gls@suffixFF" :length 2 :ignore-end)}}%
3017 \fi

```

Specify how to format locations.

```

3018 \write\glswrite{^^J; define format to use for locations^^J}%
3019 \write\glswrite{\@xdylocref}%

```

Specify how to separate letter groups.

```

3020 \write\glswrite{^^J; define letter group list format^^J}%
3021 \write\glswrite{(markup-letter-group-list
3022     :sep \string"\string\glsgroupskip\string~n\string")}%

```

Define letter group headings.

```

3023 \write\glswrite{^^J; letter group headings^^J}%
3024 \write\glswrite{(markup-letter-group
3025     :open-head \string"\string\glsgroupheading
3026     \glsopenbrace\string"^^J\space\space\space
3027     :close-head \string"\glsclosebrace\string")}%

```

Define additional letter groups.

```

3028 \write\glswrite{^^J; additional letter groups^^J}%
3029 \write\glswrite{\@xdylettergroups}%

```

Define additional sort rules

```

3030 \write\glswrite{^^J; additional sort rules^^J}
3031 \write\glswrite{\@xdysortrules}%

```

Close the style file

```

3032 \closeout\glswrite

```

Suppress any further calls.

```

3033 \let\writeist\relax
3034 }
3035 \else

```

Code to use if makeindex is required.

```

3036 \edef\@gls@actualchar{\string?}
3037 \edef\@gls@encapchar{\string|}
3038 \edef\@gls@levelchar{\string!}
3039 \edef\@gls@quotechar{\string"}
3040 \def\writeist{\relax
3041 \openout\glswrite=\istfilename
3042 \write\glswrite{\expandafter\@gobble\string\% makeindex style file
3043     created by the glossaries package}
3044 \write\glswrite{\expandafter\@gobble\string\% for document
3045     '\jobname' on \the\year-\the\month-\the\day}

```

```

3046 \write\glswrite{actual '\@gls@actualchar'}
3047 \write\glswrite{encap '\@gls@encapchar'}
3048 \write\glswrite{level '\@gls@levelchar'}
3049 \write\glswrite{quote '\@gls@quotechar'}
3050 \write\glswrite{keyword \string"\string\glossaryentry\string"}
3051 \write\glswrite{preamble \string"\string\glossarysection[\string
3052 \glossarytoctitle]{\string\glossarytitle}\string
3053 \glossarypreamble\string\n\string\begin{theglossary}\string
3054 \glossaryheader\string\n\string"}
3055 \write\glswrite{postamble \string"\string%\string\n\string
3056 \end{theglossary}\string\glossarypostamble\string\n
3057 \string"}
3058 \write\glswrite{group_skip \string"\string\glsgroupskip\string\n
3059 \string"}
3060 \write\glswrite{item_0 \string"\string%\string\n\string"}
3061 \write\glswrite{item_1 \string"\string%\string\n\string"}
3062 \write\glswrite{item_2 \string"\string%\string\n\string"}
3063 \write\glswrite{item_01 \string"\string%\string\n\string"}
3064 \write\glswrite{item_x1
3065 \string"\string\relax \string\glsresetentrylist\string\n
3066 \string"}
3067 \write\glswrite{item_12 \string"\string%\string\n\string"}
3068 \write\glswrite{item_x2
3069 \string"\string\relax \string\glsresetentrylist\string\n
3070 \string"}

3071 \write\glswrite{delim_0 \string"\string\{\string
3072 \glossaryentrynumbers\string\{\string\relax \string"}
3073 \write\glswrite{delim_1 \string"\string\{\string
3074 \glossaryentrynumbers\string\{\string\relax \string"}
3075 \write\glswrite{delim_2 \string"\string\{\string
3076 \glossaryentrynumbers\string\{\string\relax \string"}
3077 \write\glswrite{delim_t \string"\string\}\string\}\string"}
3078 \write\glswrite{delim_n \string"\string\delimN \string"}
3079 \write\glswrite{delim_r \string"\string\delimR \string"}
3080 \write\glswrite{headings_flag 1}
3081 \write\glswrite{heading_prefix
3082 \string"\string\glsgroupheading\string\{\string"}
3083 \write\glswrite{heading_suffix
3084 \string"\string\}\string\relax
3085 \string\glsresetentrylist \string"}
3086 \write\glswrite{symhead_positive \string"glssymbols\string"}
3087 \write\glswrite{numhead_positive \string"glssymbols\string"}
3088 \write\glswrite{page_compositor \string"glscpositor\string"}
3089 \@gls@escbsdq\gls@suffixF
3090 \@gls@escbsdq\gls@suffixFF
3091 \ifx\gls@suffixF\empty
3092 \else
3093 \write\glswrite{suffix_2p \string"\gls@suffixF\string"}
3094 \fi

```

```

3095 \ifx\gls@suffixFF\@empty
3096 \else
3097 \write\glswrite{suffix_3p \string"\gls@suffixFF\string"}
3098 \fi
3099 \closeout\glswrite
3100 \let\writeist\relax
3101 }
3102 \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`

```

3103 \newcommand{\noist}{%
  Update attributes list
3104 \@gls@addpredefinedattributes
3105 \let\writeist\relax
3106 }

```

`\@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`

```

3107 \newcommand*{\@makeglossary}[1]{%
3108 \ifglossaryexists{#1}%
3109 {%

```

Only create a new write if `savewrites=false` otherwise create a token to collect the information.

```

3110 \ifglssavewrites
3111 \expandafter\newtoks\csname glo@#1@filetok\endcsname
3112 \else
3113 \expandafter\newwrite\csname glo@#1@file\endcsname
3114 \expandafter\@glsopenfile\csname glo@#1@file\endcsname{#1}%
3115 \fi
3116 \@gls@renewglossary
3117 \writeist
3118 }%
3119 {%
3120 \PackageError{glossaries}%

```

```

3121 {Glossary type ‘#1’ not defined}%
3122 {New glossaries must be defined before using \string\makeglossary}%
3123 }%
3124 }

```

`\@glsopenfile` Open write file associated with the given glossary.

```

3125 \newcommand*{\@glsopenfile}[2]{%
3126 \immediate\openout#1=\jobname.\csname @glotype@#2@out\endcsname
3127 \PackageInfo{glossaries}{Writing glossary file
3128 \jobname.\csname @glotype@#2@out\endcsname}%
3129 }

```

`\nomakeglossaries` Issue warning that `\makeglossaries` hasn't been used.

```

3130 \newcommand*{\warn@nomakeglossaries}{%
3131 \GlossariesWarningNoLine{\string\makeglossaries\space
3132 hasn't been used,^^Jthe glossaries will not be updated}%
3133 }

```

`\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`

```

3134 \newcommand*{\makeglossaries}{%
    Write the name of the style file to the aux file (needed by makeglossaries)
3135 \protected@write\@auxout{}{\string\@istfilename{\istfilename}}%
3136 \protected@write\@auxout{}{\string\@glsorder{\glsorder}}%
    Iterate through each glossary type and activate it.
3137 \@for\@glo@type:=\@glo@types\do{%
3138 \ifthenelse{\equal{\@glo@type}{}}{}{}%
3139 \@makeglossary{\@glo@type}}%
3140 }%

```

New glossaries must be created before `\makeglossaries` so disable `\newglossary`.

```

3141 \renewcommand*\newglossary[4][]{%
3142 \PackageError{glossaries}{New glossaries
3143 must be created before \string\makeglossaries}{You need
3144 to move \string\makeglossaries\space after all your
3145 \string\newglossary\space commands}}%

```

Any subsequent instances of this command should have no effect

```

3146 \let\@makeglossary\relax
3147 \let\makeglossary\relax
3148 \let\makeglossaries\relax

```

Disable all commands that have no effect after `\makeglossaries`

```

3149 \@disable@onlypremakeg

```



```

    Suppress warning about no \makeglossaries
3150 \let\warn@nomakeglossaries\relax

    Declare list parser for \glsdisplaynumberlist
3151 \ifglssavenumberlist
3152   \edef\@gls@dodolistparser{\noexpand\DeclareListParser
3153     {\noexpand\glsnumlistparser}{\delimN}}}%
3154   \@gls@dodolistparser
3155 \fi
3156 }

```

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`

```

3157 \let\makeglossary\makeglossaries

    If \makeglossaries hasn't been used, issue a warning. Also issue a warning
    if neither \printglossaries nor \printglossary have been used.
3158 \AtEndDocument{%
3159   \warn@nomakeglossaries
3160   \warn@noprintglossary
3161 }

```

1.13 Writing information to associated files

`\glswrite` The write used for style file also used for all other output files if `savewrites=true`.

```

3162 \newwrite\glswrite

```

`\istfile` Deprecated.

```

3163 \def\istfile{\glswrite}

```

At the end of the document, the files should be created if `savewrites=true`.

```

3164 \AtEndDocument{%
3165   \glswritefiles
3166 }

```

`\glswritefiles` Only write the files if `savewrites=true`

```

3167 \ifglssavewrites
3168   \newcommand*{\glswritefiles}{%

```

Iterate through all the glossaries

```

3169   \forallglossaries{\@glo@type}{%

```

Check for empty glossaries (patch provided by Patrick Häcker)

```

3170     \ifcsundef{glo@\@glo@type @filetok}%
3171     {%
3172       \def\gls@tmp{%

```

```

3173     }%
3174     {%
3175         \edef\gls@tmp{\expandafter\the
3176             \csname glo@\@glo@type @filetok\endcsname}%
3177     }%
3178     \ifx\gls@tmp\@empty
3179         \ifx\@glo@type\glsdefaulttype
3180             \GlossariesWarningNoLine{Glossary '@glo@type' has no
3181                 entries.^^JRemember to use package option 'nomain' if
3182 you
3183                 don't want to^^Juse the main glossary}%
3184         \else
3185             \GlossariesWarningNoLine{Glossary '@glo@type' has no
3186                 entries}%
3187         \fi
3188     \else
3189         \@glsopenfile{\glswrite}{\@glo@type}%
3190         \immediate\write\glswrite{%
3191             \expandafter\the
3192                 \csname glo@\@glo@type @filetok\endcsname}%
3193         \immediate\closeout\glswrite
3194     \fi
3195 }%
3196 }
3197 \else
3198     \let\glswritefiles\relax
3199 \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`

```

3200 \renewcommand*{\glossary}[1][\glsdefaulttype]{%
3201     \@glossary[#1]%
3202 }

```

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`

```

3203 \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`

```

3204 \newcommand{\@gls@renewglossary}{%
3205   \gdef\@glossary[##1]{\@bsphack\begingroup\@wrglossary{##1}}%
3206   \let\@gls@renewglossary\@empty
3207 }

```

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`

```

3208 \renewcommand*{\@wrglossary}[2]{%
3209   \ifglssavewrites
3210     \protected@edef\@gls@tmp{\the\csname glo@#1@filetok\endcsname#2}%
3211     \expandafter\global\expandafter\csname glo@#1@filetok\endcsname
3212       \expandafter{\@gls@tmp^^J}%
3213   \else
3214     \ifcsdef{glo@#1@file}%
3215     {%
3216       \expandafter\protected@write\csname glo@#1@file\endcsname{}{#2}%
3217     }%
3218     {%
3219       \GlossariesWarning{No file defined for glossary ‘#1’}%
3220     }%
3221   \fi
3222   \endgroup\@esphack
3223 }

```

`\@do@wrglossary`

```

3224 \newcommand*{\@do@wrglossary}[1]{%
3225   \ifglsindexonlyfirst
3226     \ifglsused{#1}{\@do@wrglossary{#1}}%
3227   \else
3228     \@do@wrglossary{#1}%
3229   \fi
3230 }

```

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.

```

3231 \newcommand*{\@do@wrglossary}[1]{%

```

Get the location and escape any special characters

```

3232   \protected@edef\@glslocref{\theglsentrycounter}%
3233   \@gls@checkmkidxchars\@glslocref

```

Check if the hyper-location is the same as the location and set the hyper prefix.

```

3234   \expandafter\ifx\theHglentrycounter\theglsentrycounter
3235   \def\@glo@counterprefix{%

```

```

3236 \else
3237   \protected@edef\@glsHlocref{\theHglentrycounter}%
3238   \@gls@checkmkidxchars\@glsHlocref
3239   \edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix
3240     {\@glslocref}{\@glsHlocref}%
3241   }%
3242   \@do@gls@getcounterprefix
3243 \fi

```

Determine whether to use xindy or makeindex syntax

```

3244 \ifglxindy

```

Need to determine if the formatting information starts with a (or) indicating a range.

```

3245 \expandafter\@glo@check@mkidxrangechar\@glsnumberformat\@nil
3246 \def\@glo@range{}%
3247 \expandafter\if\@glo@prefix(\relax
3248   \def\@glo@range{:open-range}%
3249 \else
3250   \expandafter\if\@glo@prefix)\relax
3251   \def\@glo@range{:close-range}%
3252 \fi
3253 \fi

```

Write to the glossary file using xindy syntax.

```

3254 \glossary[\csname glo@#1@type\endcsname]{%
3255   (indexentry :tkey (\csname glo@#1@index\endcsname)
3256     :locref \string{\@glo@counterprefix}{\theHglentrycounter}\string" %
3257     :attr \string\@gls@counter\@glo@suffix\string"
3258     \@glo@range
3259   )
3260   }%
3261 \else

```

Convert the format information into the format required for makeindex

```

3262 \@set@glo@numformat{\@glo@numfmt}{\@gls@counter}{\@glsnumberformat}%
3263 {\@glo@counterprefix}%

```

Write to the glossary file using makeindex syntax.

```

3264 \glossary[\csname glo@#1@type\endcsname]{%
3265   \string\glossaryentry{\csname glo@#1@index\endcsname
3266     \@gls@encapchar\@glo@numfmt}{\theHglentrycounter}}%
3267 \fi
3268 }

```

`\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.)

```

3269 \newcommand*\@gls@getcounterprefix[2]{%
3270   \edef\@gls@thisloc{#1}\edef\@gls@thisHloc{#2}%
3271   \ifx\@gls@thisloc\@gls@thisHloc
3272     \def\@glo@counterprefix{%
3273   \else
3274     \def\@gls@get@counterprefix##1.#1##2\end@getprefix{%
3275       \def\@glo@tmp{##2}%
3276       \ifx\@glo@tmp\@empty
3277         \def\@glo@counterprefix{%
3278       \else
3279         \def\@glo@counterprefix{##1}%
3280       \fi
3281     }%
3282     \@gls@get@counterprefix#2.#1\end@getprefix
3283   \fi
3284 }

```

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 `[\langle tag \rangle]{\langle list \rangle}`, where `\langle tag \rangle` is a tag such as “see” and `\langle list \rangle` is a list of labels.

```

3285 \newcommand{\@do@seeglossary}[2]{%
3286   \def\@gls@xref{#2}%
3287   \@onelevel@sanitize\@gls@xref
3288   \@gls@checkmkidxchars\@gls@xref
3289   \ifglxsindy
3290     \glossary[\csname glo@#1@type\endcsname]{%
3291       (indexentry
3292         :tkey (\csname glo@#1@index\endcsname)
3293         :xref (\string"\@gls@xref\string")
3294         :attr \string"see\string"
3295       )
3296     }%
3297   \else
3298     \glossary[\csname glo@#1@type\endcsname]{%
3299       \string\glossaryentry{\csname glo@#1@index\endcsname
3300         \@gls@encapchar glsseeformat\@gls@xref}{Z}}%
3301   \fi
3302 }

```

`\@gls@fixbraces` If no optional argument is specified, list needs to be enclosed in a set of braces.

```

3303 \def\@gls@fixbraces#1#2#3\@nil{%
3304   \ifx#2[\relax
3305     \def#1{#2#3}%
3306   \else
3307     \def#1{{#2#3}}%
3308   \fi
3309 }

```

```

\glssee \glssee{<label>}{<cross-ref list>}
3310 \newcommand*{\glssee}[3][\seename]{%
3311 \do@seeglossary{#2}{[#1]{#3}}}
3312 \newcommand*{\@glssee}[3][\seename]{%
3313 \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.

```

3314 \newcommand*{\glsseeformat}[3][\seename]{\emph{#1} \glsseelist{#2}}

```

`\glsseelist` `\glsseelist{<list>}` formats list of entry labels.

```

3315 \newcommand*{\glsseelist}[1]{%
    If there is only one item in the list, set the last separator to do nothing.
3316 \let\@gls@dolast\relax
    Don't display separator on the first iteration of the loop
3317 \let\@gls@donext\relax
    Iterate through the labels
3318 \@for\@gls@thislabel:=#1\do{%
    Check if on last iteration of loop
3319 \ifx\@xfor@nextelement\@nnil
3320 \@gls@dolast
3321 \else
3322 \@gls@donext
3323 \fi
    display the entry for this label
3324 \glsseeitem{\@gls@thislabel}%
    Update separators
3325 \let\@gls@dolast\glsseelastsep
3326 \let\@gls@donext\glsseesep
3327 }%
3328 }

```

`\glsseelastsep` Separator to use between penultimate and ultimate entries in a cross-referencing list.

```

3329 \newcommand*{\glsseelastsep}{\space\andname\space}

```

`\glsseesep` Separator to use between entries in a cross-referencing list.

```

3330 \newcommand*{\glsseesep}{, }

```

`\glsseeitem` `\glsseeitem{<label>}` formats individual entry in a cross-referencing list.

```

3331 \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.)

```

3332 \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.

```
3333 \newcommand*{\gls@save@numberlist}[1]{%
3334   \ifglssavenumberlist
3335     \toks@{#1}%
3336     \edef\@do@writeaux@info{%
3337       \noexpand\csgdef{glo@\glscurrententrylabel @numberlist}{\the\toks@}%
3338     }%
3339     \@onelevel@sanitize\@do@writeaux@info
3340     \protected@write\@auxout{}\{\@do@writeaux@info}%
3341   \fi
3342 }
```

`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.)

```
3343 \def\warn@noprintglossary{%
3344   \GlossariesWarningNoLine{No \string\printglossary\space
3345     or \string\printglossaries\space
3346     found.^^JThis document will not have a glossary}%
3347 }
```

`\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.

```
3348 \ifcsundef{printglossary}{}%
3349 {%
```

If `\printglossary` is already defined, issue a warning and undefine it.

```
3350   \GlossariesWarning{Overriding \string\printglossary}%
3351   \undef\printglossary
3352 }
```

`\printglossary` has an optional argument. The default value is to set the glossary type to the main glossary.

```
3353 \newcommand*{\printglossary}[1][type=\glstypedefaulttype]{%
```

If `xindy` is being used, need to find the root language for `makeglossaries` to pass to `xindy`.

```
3354   \ifglsxindy\findrootlanguage\fi
```

Set up defaults.

```
3355 \def\@glo@type{\glsdefaulttype}%
3356 \def\glossarytitle{\csname @glo@type @title\endcsname}%
3357 \let\org@glossarytitle\glossarytitle
3358 \def\@glossarystyle{}%
3359 \def\gls@dotocitle{\glssettocitle{\@glo@type}}%
```

Store current value of `\glossaryentrynumbers`. (This may be changed via the optional argument)

```
3360 \let\@org@glossaryentrynumbers\glossaryentrynumbers
```

Localise the effects of the optional argument

```
3361 \bgroup
```

Determine settings specified in the optional argument.

```
3362 \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)

```
3363 \ifx\glossarytitle\org@glossarytitle
3364 \else
3365 \expandafter\let\csname @glo@type @title\endcsname
3366 \glossarytitle
3367 \fi
```

Allow a high-level user command to indicate the current glossary

```
3368 \let\currentglossary\@glo@type
```

Enable individual number lists to be suppressed.

```
3369 \let\org@glossaryentrynumbers\glossaryentrynumbers
3370 \let\glsnonextpages\@glsnonextpages
```

Enable individual number list to be activated:

```
3371 \let\glsnextpages\@glsnextpages
```

Enable suppression of description terminators.

```
3372 \let\nopostdesc\@nopostdesc
```

Set up the entry for the TOC

```
3373 \gls@dotocitle
```

Set the glossary style

```
3374 \@glossarystyle
```

added a way to fetch the current entry label:

```
3375 \let\gls@org@glossaryentryfield\glossaryentryfield
3376 \let\gls@org@glossarysubentryfield\glossarysubentryfield
3377 \renewcommand{\glossaryentryfield}[1]{%
3378 \gdef\glscurrententrylabel{##1}%
3379 \gls@org@glossaryentryfield{##1}%
3380 }%
3381 \renewcommand{\glossarysubentryfield}[2]{%
3382 \gdef\glscurrententrylabel{##2}%
```



```

3383     \gls@org@glossarysubentryfield{##1}{##2}%
3384 }%

```

Some macros may end up being expanded into internals in the glossary, so need to make @ a letter.

```

3385     \makeatletter

```

Input the glossary file, if it exists.

```

3386     \input@{\jobname.\csname @glo@type @in\endcsname}%

```

If the glossary file doesn't exist, do \null. (This ensures that the page is shipped out and all write commands are done.) This might produce an empty page, but at this point the document isn't complete, so it shouldn't matter.

```

3387     \IfFileExists{\jobname.\csname @glo@type @in\endcsname}%
3388     {}%
3389     {\null}%

```

If xindy is being used, need to write the language dependent information to the .aux file for makeglossaries.

```

3390     \ifglxindy
3391         \ifcsundef{@xdy@\@glo@type @language}%
3392         {%
3393             \protected@write\@auxout{}{%
3394                 \string\@xdylanguage{\@glo@type}{\@xdy@main@language}}%
3395             }%
3396             {%
3397                 \protected@write\@auxout{}{%
3398                     \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type
3399                         @language\endcsname}}%
3400                 }%
3401                 \protected@write\@auxout{}{%
3402                     \string\@gls@codepage{\@glo@type}{\@gls@codepage}}%
3403             }%
3404     \egroup

```

Reset \glossaryentrynumbers

```

3405     \global\let\glossaryentrynumbers\@org@glossaryentrynumbers

```

Suppress warning about no \printglossary

```

3406     \global\let\warn@noprintglossary\relax
3407 }

```

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`

```
3408 \newcommand*{\printglossaries}{%
3409   \forallglossaries{\@glo@type}{\printglossary[type=\@glo@type]}%
3410 }
```

The keys that can be used in the optional argument to `\printglossary` are as follows: The type key sets the glossary type.

```
3411 \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`.

```
3412 \define@key{printgloss}{title}{\def\glossarytitle{#1}}
```

The toctitle sets the text used for the relevant entry in the table of contents.

```
3413 \define@key{printgloss}{toctitle}{\def\glossarytoctitle{#1}%
3414 \let\gls@dotoc@title\relax
3415 }
```

The style key sets the glossary style (but only for the given glossary).

```
3416 \define@key{printgloss}{style}{%
3417   \ifcsundef{@glsstyle@#1}%
3418   {%
3419     \PackageError{glossaries}%
3420     {Glossary style ‘#1’ undefined}{}%
3421   }%
3422   {%
3423     \def\@glossarystyle{\csname @glsstyle@#1\endcsname}%
3424   }%
3425 }
```

The numberedsection key determines if this glossary should be in a numbered section.

```
3426 \define@choicekey{printgloss}{numberedsection}[\val\nr]{%
3427 false,nolabel,autolabel}[nolabel]{%
3428 \ifcase\nr\relax
3429   \renewcommand*{\@glossarysecstar}{*}%
3430   \renewcommand*{\@glossaryseclabel}{}%
3431 \or
3432   \renewcommand*{\@glossarysecstar}{}%
3433   \renewcommand*{\@glossaryseclabel}{}%
3434 \or
3435   \renewcommand*{\@glossarysecstar}{}%
3436   \renewcommand*{\@glossaryseclabel}{\label{\glsautoprefix\@glo@type}}%
3437 \fi}
```

The nonumberlist key determines if this glossary should have a number list.

```
3438 \define@boolkey{printgloss}[gls]{nonumberlist}[true]{%
3439 \ifglsnonumberlist
3440   \def\glossaryentrynumbers##1{}%
3441 \else
3442   \def\glossaryentrynumbers##1{##1}%
3443 }
```

3443 \fi}

`\@glsnonextpages` Suppresses the next number list only. Global assignments required as it may not occur in the same level of grouping as the next numberlist. (For example, if `\glsnonextpages` is place in the entry's description and 3 column tabular style glossary is used.) `\org@glossaryentrynumbers` needs to be set at the start of each glossary, in the event that `\glossaryentrynumber` is redefined.

```
3444 \newcommand*{\@glsnonextpages}{%
3445   \gdef\glossaryentrynumbers##1{%
3446     \glsresetentrylist
3447   }%
3448 }
```

`\@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.

```
3449 \newcommand*{\@glsnextpages}{%
3450   \gdef\glossaryentrynumbers##1{%
3451     ##1\glsresetentrylist}}
```

`\glsresetentrylist` Resets `\glossaryentrynumbers`

```
3452 \newcommand*{\glsresetentrylist}{%
3453   \global\let\glossaryentrynumbers\org@glossaryentrynumbers}
```

`\glsnonextpages` Outside of `\printglossary` this does nothing.

```
3454 \newcommand*{\glsnonextpages}{}%
```

`\glsnextpages` Outside of `\printglossary` this does nothing.

```
3455 \newcommand*{\glsnextpages}{}%
```

`glossaryentry` If the `entrycounter` package option has been used, define a counter to number each level 0 entry.

```
3456 \ifglentrycounter
3457   \ifx\@gls@counterwithin\@empty
3458     \newcounter{glossaryentry}
3459   \else
3460     \newcounter{glossaryentry}[\@gls@counterwithin]
3461   \fi
3462   \def\theHglossaryentry{\currentglossary.\theglossaryentry}
3463 \fi
```

`glossarysubentry` If the `subentrycounter` package option has been used, define a counter to number each level 1 entry.

```
3464 \ifglsubentrycounter
3465   \ifglentrycounter
```

```

3466 \newcounter{glossarysubentry}[glossaryentry]
3467 \else
3468 \newcounter{glossarysubentry}
3469 \fi
3470 \def\theHglossarysubentry{\currentglssubentry.\theglossarysubentry}
3471 \fi

```

`\resetsubentrycounter` Resets the glossarysubentry counter.

```

3472 \ifglssubentrycounter
3473 \newcommand*{\glsresetsubentrycounter}{%
3474 \setcounter{glossarysubentry}{0}%
3475 }
3476 \else
3477 \newcommand*{\glsresetsubentrycounter}{}
3478 \fi

```

`\resetentrycounter` Resets the glossaryentry counter.

```

3479 \ifglentrycounter
3480 \newcommand*{\glsresetentrycounter}{%
3481 \setcounter{glossaryentry}{0}%
3482 }
3483 \else
3484 \newcommand*{\glsresetentrycounter}{}
3485 \fi

```

`\glsstepentry` Advance the glossaryentry counter if in use. The argument is the label associated with the entry.

```

3486 \ifglentrycounter
3487 \newcommand*{\glsstepentry}[1]{%
3488 \refstepcounter{glossaryentry}%
3489 \label{glsentry-#1}%
3490 }
3491 \else
3492 \newcommand*{\glsstepentry}[1]{}
3493 \fi

```

`\glsstepsubentry` Advance the glossarysubentry counter if in use. The argument is the label associated with the subentry.

```

3494 \ifglssubentrycounter
3495 \newcommand*{\glsstepsubentry}[1]{%
3496 \def\currentglssubentry{#1}%
3497 \refstepcounter{glossarysubentry}%
3498 \label{glsentry-#1}%
3499 }
3500 \else
3501 \newcommand*{\glsstepsubentry}[1]{}
3502 \fi

```

`\glsrefentry` Reference the entry or sub-entry counter if in use, otherwise just do `\gls`.

```
3503 \ifglentrycounter
3504   \newcommand*{\glsrefentry}[1]{\ref{glentry-#1}}
3505 \else
3506   \ifglssubentrycounter
3507     \newcommand*{\glsrefentry}[1]{\ref{glentry-#1}}
3508   \else
3509     \newcommand*{\glsrefentry}[1]{\gls{#1}}
3510   \fi
3511 \fi
```

`lentrycounterlabel` Defines how to display the glossaryentry counter.

```
3512 \ifglentrycounter
3513   \newcommand*{\glsentrycounterlabel}{\theglossaryentry.\space}
3514 \else
3515   \newcommand*{\glsentrycounterlabel}{}
3516 \fi
```

`ubentrycounterlabel` Defines how to display the glossarysubentry counter.

```
3517 \ifglssubentrycounter
3518   \newcommand*{\glssubentrycounterlabel}{\theglossarysubentry}\space}
3519 \else
3520   \newcommand*{\glssubentrycounterlabel}{}
3521 \fi
```

`\glstryitem` Step and display glossaryentry counter, if appropriate.

```
3522 \ifglentrycounter
3523   \newcommand*{\glstryitem}[1]{%
3524     \glsstepentry{#1}\glsentrycounterlabel
3525   }
3526 \else
3527   \newcommand*{\glstryitem}[1]{\glsresetsubentrycounter}
3528 \fi
```

`\glssubstryitem` Step and display glossarysubentry counter, if appropriate.

```
3529 \ifglssubentrycounter
3530   \newcommand*{\glssubstryitem}[1]{%
3531     \glsstepsubentry{#1}\glssubentrycounterlabel
3532   }
3533 \else
3534   \newcommand*{\glssubstryitem}[1]{}
3535 \fi
```

`theglossary` If the `theglossary` environment has already been defined, a warning will be issued. This environment should be redefined by glossary styles.

```
3536 \ifcsundef{theglossary}%
3537 {%
3538   \newenvironment{theglossary}{}{}}%
```

```

3539 }%
3540 {%
3541   \GlossariesWarning{overriding ‘theglossary’ environment}%
3542   \renewenvironment{theglossary}{}{}%
3543 }

```

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`

```

3544 \newcommand*{\glossaryheader}{}

```

`\glstarget` `\glstarget{<label>}{<name>}`

Provide user interface to `\@glstarget` to make it easier to modify the glossary style in the document.

```

3545 \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>`.

```

3546 \newcommand*{\glossaryentryfield}[5]{%
3547 \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.)

```

3548 \newcommand*{\glossarysubentryfield}[6]{%
3549 \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`

```
3550 \newcommand*{\glsgroupskip}{}{}
```

Each of the 28 glossary groups described above is preceded by a group heading. This is formatted by the command `\glsgroupheading` which takes one argument which is the *label* assigned to that group (not the title). The corresponding labels are: `glssymbols`, `glsnumbers`, A, ..., Z. Glossary styles must redefine this command. (In between groups, `\glsgroupheading` comes immediately after `\glsgroupskip`.)

`\glsgroupheading`

```
3551 \newcommand*{\glsgroupheading}[1]{}{}
```

It is possible to “trick” `makeindex` into treating entries as though they belong to the same group, even if the terms don’t start with the same letter, by modifying the sort key. For example, all entries belonging to one group could be defined so that the sort key starts with an a, while entries belonging to another group could be defined so that the sort key starts with a b, and so on. If you want each group to have a heading, you would then need to modify the translation control sequences `\glsgetgrouptitle` and `\glsgetgrouplabel` so that the label is translated into the required title (and vice-versa).

`\glsgetgrouptitle{<label>}`

This command produces the title for the glossary group whose label is given by `<label>`. By default, the group labelled `glssymbols` produces `\glssymbolsgroupname`, the group labelled `glsnumbers` produces `\glsnumbersgroupname` and all the other groups simply produce their label. As mentioned above, the group labels are: `glssymbols`, `glsnumbers`, A, ..., Z. If you want to redefine the group titles, you will need to redefine this command.

`\glsgetgrouptitle`

```
3552 \newcommand*{\glsgetgrouptitle}[1]{%
3553   \ifcsundef{#1groupname}{#1}{\csname #1groupname\endcsname}%
3554 }
```

`\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`

```
3555 \newcommand*{\glsgetgrouplabel}[1]{%
3556   \ifthenelse{\equals{#1}{\glssymbolsgroupname}}{\glssymbols}{%
3557   \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`

```
3558 \newcommand*{\setentrycounter}[2][]{%
3559   \def\@glo@counterprefix{#1}%
3560   \ifx\@glo@counterprefix\@empty
3561     \def\@glo@counterprefix{.}%
3562   \else
3563     \def\@glo@counterprefix{.#1.}%
3564   \fi
3565   \def\glsentrycounter{#2}%
3566 }
```

The current glossary style can be set using `\glossarystyle{<style>}`.

`\glossarystyle`

```
3567 \newcommand*{\glossarystyle}[1]{%
3568   \ifcsundef{@glsstyle@#1}%
3569   {%
3570     \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
3571   }%
3572   {%
3573     \csname @glsstyle@#1\endcsname
3574   }%
3575 }
```

`\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.

```
3576 \newcommand{\newglossarystyle}[2]{%
3577   \ifcsundef{@glsstyle@#1}%
3578   {%
3579     \expandafter\def\csname @glsstyle@#1\endcsname{#2}%
3580   }%
3581   {%
3582     \PackageError{glossaries}{Glossary style ‘#1’ is already defined}{}%
3583   }%
3584 }
```

`\renewglossarystyle` Code for this macro supplied by Marco Daniel.

```
3585 \newcommand{\renewglossarystyle}[2]{%
```



```

3586 \ifcsundef{@glsstyle@#1}%
3587 {%
3588   \PackageError{glossaries}{Glossary style ‘#1’ isn’t already defined}{}%
3589 }%
3590 {%
3591   \csdef{@glsstyle@#1}{#2}%
3592 }%
3593 }

```

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`

```

3594 \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`

```

3595 \ifcsundef{hyperlink}%
3596 {%
3597   \def\glshypernumber#1{#1}%
3598 }%
3599 {%
3600   \def\glshypernumber#1{\@glshypernumber#1\nohyperpage{}}\@nil}
3601 }

```

`\@glshypernumber` This code was provided by Heiko Oberdiek to allow material to be attached to the location.

```

3602 \def\@glshypernumber#1\nohyperpage#2#3\@nil{%
3603   \ifx\#1\%
3604   \else

```

```

3605 \@delimR#1\delimR\delimR\\%
3606 \fi
3607 \ifx\\#2\\%
3608 \else
3609 #2%
3610 \fi
3611 \ifx\\#3\\%
3612 \else
3613 \@glshypernumber#3\@nil
3614 \fi
3615 }

```

\@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

```

3616 \def\@delimR#1\delimR #2\delimR #3\\{%
3617 \ifx\\#2\\%
3618 \@delimN{#1}%
3619 \else
3620 \@gls@numberlink{#1}\delimR\@gls@numberlink{#2}%
3621 \fi}

```

\@delimN displays a list of individual numbers, instead of a range:

\@delimN

```

3622 \def\@delimN#1{\@delimN#1\delimN \delimN\\}
3623 \def\@delimN#1\delimN #2\delimN#3\\{%
3624 \ifx\\#3\\%
3625 \@gls@numberlink{#1}%
3626 \else
3627 \@gls@numberlink{#1}\delimN\@gls@numberlink{#2}%
3628 \fi
3629 }

```

The following code is modified from hyperref's \@HyInd@pagelink where the name of the counter being used is given by \@gls@counter.

```

3630 \def\@gls@numberlink#1{%
3631 \begingroup
3632 \toks@={}%
3633 \@gls@removespaces#1 \@nil
3634 \endgroup}

3635 \def\@gls@removespaces#1 #2\@nil{%
3636 \toks@=\expandafter{\the\toks@#1}%
3637 \ifx\\#2\\%
3638 \edef\x{\the\toks@}%
3639 \ifx\x\empty
3640 \else

```

```

3641     \hyperlink{\glentrycounter\glo@counterprefix\the\toks@}%
3642             {\the\toks@}%
3643     \fi
3644 \else
3645     \@gls@ReturnAfterFi{%
3646         \@gls@removespaces#2\@nil
3647     }%
3648 \fi
3649 }
3650 \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`

```
3651 \newcommand*\hyperrm[1]{\textrm{\glshypernumber{#1}}}
```

`\hypersf`

```
3652 \newcommand*\hypersf[1]{\textsf{\glshypernumber{#1}}}
```

`\hypertt`

```
3653 \newcommand*\hypertt[1]{\texttt{\glshypernumber{#1}}}
```

`\hyperbf`

```
3654 \newcommand*\hyperbf[1]{\textbf{\glshypernumber{#1}}}
```

`\hypermd`

```
3655 \newcommand*\hypermd[1]{\textmd{\glshypernumber{#1}}}
```

`\hyperit`

```
3656 \newcommand*\hyperit[1]{\textit{\glshypernumber{#1}}}
```

`\hypersl`

```
3657 \newcommand*\hypersl[1]{\textsl{\glshypernumber{#1}}}
```

`\hyperup`

```
3658 \newcommand*\hyperup[1]{\textup{\glshypernumber{#1}}}
```

`\hypersc`

```
3659 \newcommand*\hypersc[1]{\textsc{\glshypernumber{#1}}}
```

`\hyperemph`

```
3660 \newcommand*\hyperemph[1]{\emph{\glshypernumber{#1}}}
```

1.16 Acronyms

If the acronym package option is used, a new glossary called acronym is created

```
3661 \ifglsacronym
3662   \newglossary[alg]{acronym}{acr}{acn}{\acronymname}
      and \acronymtype is set to the name of this new glossary.
3663   \renewcommand*{\acronymtype}{acronym}
3664 \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.

```
3665 \newcommand{\oldacronym}[4][\gls@label]{%
3666   \def\gls@label{#2}%
3667   \newacronym[#4]{#1}{#2}{#3}%
3668   \ifcsundef{xspace}%
3669     {%
3670       \expandafter\edef\csname#1\endcsname{%
3671         \noexpand\@ifstar{\noexpand\Gls{#1}}{\noexpand\gls{#1}}}%
3672       }%
3673     }%
3674     {%
3675       \expandafter\edef\csname#1\endcsname{%
3676         \noexpand\@ifstar{\noexpand\Gls{#1}\noexpand\xspace}{%
3677           \noexpand\gls{#1}\noexpand\xspace}%
3678         }%
3679       }%
3680 }
```

```
\newacronym[⟨key-val list⟩]{⟨label⟩}{⟨abbrv⟩}{⟨long⟩}
```

This is a quick way of defining acronyms, all it does is call `\newglossaryentry` with the appropriate values. It sets the glossary type to `\acronymtype` which

will be acronym if the package option acronym has been used, otherwise it will be the default glossary. Since `\newacronym` merely calls `\newglossaryentry`, the acronym is treated like any other glossary entry.

If you prefer a different format, you can redefine `\newacronym` as required. The optional argument can be used to override any of the settings.

This is just a stub. It's redefined by commands like `\SetDefaultAcronymStyle`.

`\newacronym`

```
3681 \newcommand{\newacronym}[4] [] {}
```

Set up some convenient short cuts. These need to be changed if `\newacronym` is changed (or if the description key is changed).

`\acrpluralsuffix`

Plural suffix used by `\newacronym`. This just defaults to `\glspluralsuffix` but is changed to include `\textup` if the smallcaps option is used, so that the suffix doesn't appear in small caps as it doesn't look right. For example, ABCS looks as though the "s" is part of the acronym, but ABCs looks as though the "s" is a plural suffix. Since the entire text abcs is set in `\textsc`, `\textup` is needed to cancel it out.

```
3682 \newcommand*{\acrpluralsuffix}{\glspluralsuffix}
```

The following are defined for compatibility with version 2.07 and earlier.

`\glsshortkey`

```
3683 \newcommand*{\glsshortkey}{short}
```

`\glsshortpluralkey`

```
3684 \newcommand*{\glsshortpluralkey}{shortplural}
```

`\glslongkey`

```
3685 \newcommand*{\glslongkey}{long}
```

`\glslongpluralkey`

```
3686 \newcommand*{\glslongpluralkey}{longplural}
```

`\acrfull` Full form of the acronym.

```
3687 \newrobustcmd*{\acrfull}{%
```

```
3688 \ifstar\s@acrfull\ns@acrfull
```

```
3689 }
```

```
3690 \newcommand*\s@acrfull[2] [] {%
```

```
3691 \new@ifnextchar[{\@acrfull{hyper=false,#1}{#2}}%
```

```
3692 {\@acrfull{hyper=false,#1}{#2} [] }%
```

```
3693 }
```

```
3694 \newcommand*\ns@acrfull[2] [] {%
```

```
3695 \new@ifnextchar[{\@acrfull{#1}{#2}}%
```

```
3696 {\@acrfull{#1}{#2} [] }%
```

```
3697 }
```

Low-level macro:

```
3698 \def\@acrfull#1#2[#3]{%
3699   \acrlinkfullformat{\@acrlong}{\@acrshort}{#1}{#2}{#3}%
3700 }
```

```
\acrlinkfullformat  Format for full links like \acrfull.  Syntax: \acrlinkfullformat{<long
                    cs>}{<short cs>}{<options>}{<label>}{<insert>}
3701 \newcommand{\acrlinkfullformat}[5]{%
3702   \acrfullformat{#1{#3}{#4}[#5]}{#2{#3}{#4}[]}%
3703 }
```

\acrfullformat Default full form is <long> (<short>).

```
3704 \newcommand{\acrfullformat}[2]{#1\space(#2)}
```

Default format for full acronym

\Acrfull

```
3705 \newrobustcmd*{\Acrfull}{%
3706   \@ifstar\s@Acrfull\@ns@Acrfull
3707 }

3708 \newcommand*\s@Acrfull[2][]{%
3709   \new@ifnextchar[{\@Acrfull{hyper=false,#1}{#2}}%
3710     {\@Acrfull{hyper=false,#1}{#2}[]}%
3711 }
3712 \newcommand*\ns@Acrfull[2][]{%
3713   \new@ifnextchar[{\@Acrfull{#1}{#2}}%
3714     {\@Acrfull{#1}{#2}[]}%
3715 }
```

Low-level macro:

```
3716 \def\@Acrfull#1#2[#3]{%
3717   \acrlinkfullformat{\@Acrlong}{\@acrshort}{#1}{#2}{#3}%
3718 }
```

\ACRfull

```
3719 \newrobustcmd*{\ACRfull}{%
3720   \@ifstar\s@ACRfull\@ns@ACRfull
3721 }

3722 \newcommand*\s@ACRfull[2][]{%
3723   \new@ifnextchar[{\@ACRfull{hyper=false,#1}{#2}}%
3724     {\@ACRfull{hyper=false,#1}{#2}[]}%
3725 }
3726 \newcommand*\ns@ACRfull[2][]{%
3727   \new@ifnextchar[{\@ACRfull{#1}{#2}}%
3728     {\@ACRfull{#1}{#2}[]}%
3729 }
```

Low-level macro:

```
3730 \def\@ACRfull#1#2[#3]{%
3731   \acrlinkfullformat{\@ACRlong}{\@ACRshort}{#1}{#2}{#3}%
3732 }
```

Plural:

\acrfullpl

```
3733 \newrobustcmd*\acrfullpl{%
3734   \@ifstar\s@acrfullpl\ns@acrfullpl
3735 }

3736 \newcommand*\s@acrfullpl[2][{}]{%
3737   \new@ifnextchar[{\@acrfullpl{hyper=false,#1}{#2}}%
3738     {\@acrfullpl{hyper=false,#1}{#2}[]}%
3739 }
3740 \newcommand*\ns@acrfullpl[2][{}]{%
3741   \new@ifnextchar[{\@acrfullpl{#1}{#2}}%
3742     {\@acrfullpl{#1}{#2}[]}%
3743 }
```

Low-level macro:

```
3744 \def\@acrfullpl#1#2[#3]{%
3745   \acrlinkfullformat{\@acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%
3746 }
```

\Acrfullpl

```
3747 \newrobustcmd*\Acrfullpl{%
3748   \@ifstar\s@Acrfullpl\ns@Acrfullpl
3749 }

3750 \newcommand*\s@Acrfullpl[2][{}]{%
3751   \new@ifnextchar[{\@Acrfullpl{hyper=false,#1}{#2}}%
3752     {\@Acrfullpl{hyper=false,#1}{#2}[]}%
3753 }
3754 \newcommand*\ns@Acrfullpl[2][{}]{%
3755   \new@ifnextchar[{\@Acrfullpl{#1}{#2}}%
3756     {\@Acrfullpl{#1}{#2}[]}%
3757 }
```

Low-level macro:

```
3758 \def\@Acrfullpl#1#2[#3]{%
3759   \acrlinkfullformat{\@Acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%
3760 }
```

\ACRfullpl

```
3761 \newrobustcmd*\ACRfullpl{%
3762   \@ifstar\s@ACRfullpl\ns@ACRfullpl
3763 }
```

```

3764 \newcommand*\s@ACRfullp1[2] [] {%
3765   \new@ifnextchar[{\@ACRfullp1{hyper=false,#1}{#2}}{%
3766     {\@ACRfullp1{hyper=false,#1}{#2} []}%
3767 }
3768 \newcommand*\ns@ACRfullp1[2] [] {%
3769   \new@ifnextchar[{\@ACRfullp1{#1}{#2}}{%
3770     {\@ACRfullp1{#1}{#2} []}%
3771 }

```

Low-level macro:

```

3772 \def\@ACRfullp1#1#2[#3] {%
3773   \acrlinkfullformat{\@ACRlongp1}{\@ACRshortp1}{#1}{#2}{#3}%
3774 }

```

1.17 Predefined acronym styles

`\acronymfont` This is only used with the additional acronym styles:

```

3775 \newcommand{\acronymfont}[1]{#1}

```

`\firstacronymfont` This is only used with the additional acronym styles:

```

3776 \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.

```

3777 \newcommand*\acrnameformat[2]{\acronymfont{#1}}

```

Define some tokens used by `\newacronym`:

`\glskeylisttok`

```

3778 \newtoks\glskeylisttok

```

`\glslabeltok`

```

3779 \newtoks\glslabeltok

```

`\glsshorttok`

```

3780 \newtoks\glsshorttok

```

`\glslongtok`

```

3781 \newtoks\glslongtok

```

`\newacronymhook` Provide a hook for `\newacronym`:

```

3782 \newcommand*\newacronymhook{}

```

`AcronymDisplayStyle` Sets the default acronym display style for given glossary.

```

3783 \newcommand*\SetDefaultAcronymDisplayStyle[1] {%
3784   \defglsdisplay[#1]{##1##4}%
3785   \defglsdisplayfirst[#1]{##1##4}%
3786 }

```


`DefaultNewAcronymDef` Sets up the acronym definition for the default style. The information is provided by the tokens `\glslabeltok`, `\glsshorttok`, `\glslongtok` and `\glskeylisttok`.

```

3787 \newcommand*\DefaultNewAcronymDef{%
3788   \edef\@do@newglossaryentry{%
3789     \noexpand\newglossaryentry{\the\glslabeltok}%
3790     {%
3791       type=\acronymtype,%
3792       name={\the\glsshorttok},%
3793       sort={\the\glsshorttok},%
3794       text={\the\glsshorttok},%
3795       first={\acrfullformat{\the\glslongtok}{\the\glsshorttok}},%
3796       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3797       firstplural={\acrfullformat{\noexpand\@glo@longpl}%
3798                     {\noexpand\@glo@shortpl}},%
3799       short={\the\glsshorttok},%
3800       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3801       long={\the\glslongtok},%
3802       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3803       description={\the\glslongtok},%
3804       descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%

```

Remaining options specified by the user:

```

3805     \the\glskeylisttok
3806   }%
3807 }%
3808 \@do@newglossaryentry
3809 }

```

`DefaultAcronymStyle` Set up the default acronym style:

```

3810 \newcommand*\SetDefaultAcronymStyle{%
  Set the display style:
3811   \@for\@gls@type:=\@glsacronymlists\do{%
3812     \SetDefaultAcronymDisplayStyle{\@gls@type}%
3813   }%

```

Set up the definition of `\newacronym`:

```

3814 \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).

```

3815   \ifx\@glsacronymlists\@empty
3816     \def\@glo@type{\acronymtype}%
3817     \setkeys{glossentry}{##1}%
3818     \DeclareAcronymList{\@glo@type}%
3819     \SetDefaultAcronymDisplayStyle{\@glo@type}%
3820   \fi
3821   \glskeylisttok{##1}%

```

```

3822 \glslabeltok{##2}%
3823 \glsshorttok{##3}%
3824 \glslongtok{##4}%
3825 \newacronymhook
3826 \DefaultNewAcronymDef
3827 }%
3828 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
3829 }

```

`\acrfootnote` Used by the footnote acronym styles.

```

3830 \newcommand*{\acrfootnote}[3]{\acrlinkfootnote{#1}{#2}{#3}}

```

`\acrlinkfootnote`

```

3831 \newcommand*{\acrlinkfootnote}[3]{%
3832 \footnote{\glslink{#1}{#2}{#3}}%
3833 }

```

`\acrnofootnote`

```

3834 \newcommand*{\acrnofootnote}[3]{%
3835 \footnote{#3}%
3836 }

```

`AcronymDisplayStyle` Sets the acronym display style for given glossary for the description and footnote combination.

```

3837 \newcommand*{\SetDescriptionFootnoteAcronymDisplayStyle}[1]{%
3838 \defglsdisplayfirst[#1]{%
3839 \firstacronymfont{##1}##4%
3840 \expandafter\protect\expandafter\acrfootnote\expandafter
3841 {\@gls@link@opts}{\@gls@link@label}{##3}%
3842 }%
3843 \defglsdisplay[#1]{\acronymfont{##1}##4}%
3844 }

```

`otnoteNewAcronymDef`

```

3845 \newcommand*{\DescriptionFootnoteNewAcronymDef}{%
3846 \edef\@do@newglossaryentry{%
3847 \noexpand\newglossaryentry{\the\glslabeltok}%
3848 {%
3849 type=\acronymtype,%
3850 name={\noexpand\acronymfont{\the\glsshorttok}},%
3851 sort={\the\glsshorttok},%
3852 text={\the\glsshorttok},%
3853 plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3854 short={\the\glsshorttok},%
3855 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3856 long={\the\glslongtok},%
3857 longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3858 symbol={\the\glslongtok},%

```

```

3859     symbolplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3860     \the\glskeylisttok
3861   }%
3862 }%
3863 \do@newglossaryentry
3864 }

```

footnoteAcronymStyle If a description and footnote are both required, store the long form in the symbol key. Store the short form in text key. Note that since the long form is stored in the symbol key, if you want the long form to appear in the list of acronyms, you need to use a glossary style that displays the symbol key.

```

3865 \newcommand*{\SetDescriptionFootnoteAcronymStyle}{%
3866   \renewcommand{\newacronym}[4][\]{%
3867     \ifx\@glsacronymlists\@empty
3868       \def\@glo@type{\acronymtype}%
3869       \setkeys{glossentry}{##1}%
3870       \DeclareAcronymList{\@glo@type}%
3871       \SetDescriptionFootnoteAcronymDisplayStyle{\@glo@type}%
3872     \fi
3873     \glskeylisttok{##1}%
3874     \glslabeltok{##2}%
3875     \glsshorttok{##3}%
3876     \glslongtok{##4}%
3877     \newacronymhook
3878     \DescriptionFootnoteNewAcronymDef
3879   }%

```

If footnote package option is specified, set the first use to append the long form (stored in symbol) as a footnote.

```

3880 \@for\@gls@type:=\@glsacronymlists\do{%
3881   \SetDescriptionFootnoteAcronymDisplayStyle{\@gls@type}%
3882 }%

```

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.

```

3883 \ifglsacrsmallcaps
3884   \renewcommand*{\acronymfont}[1]{\textsc{##1}}%
3885   \renewcommand*{\acrpluralsuffix}{%
3886     \textup{\glspluralsuffix}}%
3887 \else
3888   \ifglsacrsmaller
3889     \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%
3890   \fi
3891 \fi

```

Check for package option clash

```

3892 \ifglsacrdua
3893   \PackageError{glossaries}{Option clash: ‘footnote’ and ‘dua’
3894     can’t both be set}{}%

```

```

3895 \fi
3896 }%

```

AcronymDisplayStyle Sets the acronym display style for given glossary with description and dua combination.

```

3897 \newcommand*{\SetDescriptionDUAAcronymDisplayStyle}[1]{%
3898   \defglstdisplay[#1]{##1##4}%
3899   \defglstdisplayfirst[#1]{##1##4}%
3900 }

```

ionDUANewAcronymDef

```

3901 \newcommand*{\DescriptionDUANewAcronymDef}{%
3902   \edef\@do@newglossaryentry{%
3903     \noexpand\newglossaryentry{\the\glslabeltok}%
3904     {%
3905       type=\acronymtype,%
3906       name={\the\glslongtok},%
3907       sort={\the\glslongtok},%
3908       text={\the\glslongtok},%
3909       plural={\the\glslongtok\noexpand\acrpluralsuffix},%
3910       short={\the\glsshorttok},%
3911       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3912       long={\the\glslongtok},%
3913       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3914       symbol={\the\glsshorttok},%
3915       symbolplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3916       \the\glskeylisttok
3917     }%
3918   }%
3919   \@do@newglossaryentry
3920 }

```

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.

```

3921 \newcommand*{\SetDescriptionDUAAcronymStyle}{%
3922   \ifglssacrsmallcaps
3923     \PackageError{glossaries}{Option clash: 'smallcaps' and 'dua'
3924       can't both be set}{}%
3925   \else
3926     \ifglssacrsmaller
3927       \PackageError{glossaries}{Option clash: 'smaller' and 'dua'
3928         can't both be set}{}%
3929     \fi
3930   \fi
3931   \renewcommand{\newacronym}[4][[]]{%
3932     \ifx\@glsacronymlists\@empty
3933       \def\@glo@type{\acronymtype}%
3934       \setkeys{glossentry}{##1}%

```

```

3935     \DeclareAcronymList{\@glo@type}%
3936     \SetDescriptionDUAAcronymDisplayStyle{\@glo@type}%
3937     \fi
3938     \glskeylisttok{##1}%
3939     \glslabeltok{##2}%
3940     \glsshorttok{##3}%
3941     \glslongtok{##4}%
3942     \newacronymhook
3943     \DescriptionDUANewAcronymDef
3944 }%

Set display.
3945 \@for\@gls@type:=\@glsacronymlists\do{%
3946     \SetDescriptionDUAAcronymDisplayStyle{\@gls@type}%
3947 }%
3948 }%

```

AcronymDisplayStyle Sets the acronym display style for given glossary using the description setting (but not footnote or dua).

```

3949 \newcommand*\SetDescriptionAcronymDisplayStyle[1]{%
3950     \defglsdisplayfirst[#1]{%
3951         ##1##4 (\firstacronymfont{##3})}%
3952     \defglsdisplay[#1]{\acronymfont{##1}##4}%
3953 }

```

DescriptionNewAcronymDef

```

3954 \newcommand*\DescriptionNewAcronymDef{%
3955     \edef\@do@newglossaryentry{%
3956         \noexpand\newglossaryentry{\the\glslabeltok}%
3957         {%
3958             type=\acronymtype,%
3959             name={\noexpand
3960                 \acrnameformat{\the\glsshorttok}{\the\glslongtok}},%
3961             sort={\the\glsshorttok},%
3962             first={\the\glslongtok},%
3963             firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3964             text={\the\glsshorttok},%
3965             plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3966             short={\the\glsshorttok},%
3967             shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3968             long={\the\glslongtok},%
3969             longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3970             symbol={\noexpand\@glo@text},%
3971             symbolplural={\noexpand\@glo@plural},%
3972             \the\glskeylisttok}%
3973     }%
3974     \@do@newglossaryentry
3975 }

```

`DescriptionAcronymStyle` Option description is used, but not dua or footnote. Store long form in first key and short form in text and symbol key. The name is stored using `\acrnameformat` to allow the user to override the way the name is displayed in the list of acronyms.

```

3976 \newcommand*{\SetDescriptionAcronymStyle}{%
3977   \renewcommand{\newacronym}[4][\]{%
3978     \ifx\@glsacronymlists\@empty
3979       \def\@glo@type{\acronymtype}%
3980       \setkeys{glossentry}{##1}%
3981       \DeclareAcronymList{\@glo@type}%
3982       \SetDescriptionAcronymDisplayStyle{\@glo@type}%
3983     \fi
3984     \glskeylisttok{##1}%
3985     \glslabeltok{##2}%
3986     \glsshorttok{##3}%
3987     \gslongtok{##4}%
3988     \newacronymhook
3989     \DescriptionNewAcronymDef
3990   }%

```

Set display.

```

3991 \@for\@gls@type:=\@glsacronymlists\do{%
3992   \SetDescriptionAcronymDisplayStyle{\@gls@type}%
3993 }%

```

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.

```

3994 \ifglsacrsmallcaps
3995   \renewcommand{\acronymfont}[1]{\textsc{##1}}
3996   \renewcommand*{\acrpluralsuffix}{%
3997     \textup{\glspluralsuffix}}%
3998 \else
3999   \ifglsacrsmaller
4000     \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%
4001   \fi
4002 \fi
4003 }%

```

`AcronymDisplayStyle` Sets the acronym display style for given glossary with footnote setting (but not description or dua).

```

4004 \newcommand*{\SetFootnoteAcronymDisplayStyle}[1]{%
4005   \defglsdisplayfirst[#1]{%
4006     \firstacronymfont{##1}##4%
4007     \expandafter\protect\expandafter\acrfootnote\expandafter
4008       {\@gls@link@opts}{\@gls@link@label}{##2}%
4009   }%
4010   \defglsdisplay[#1]{\acronymfont{##1}##4}%
4011 }

```

otnoteNewAcronymDef

```

4012 \newcommand*{\FootnoteNewAcronymDef}{%
4013   \edef\@do@newglossaryentry{%
4014     \noexpand\newglossaryentry{\the\glslabeltok}%
4015     {%
4016       type=\acronymtype,%
4017       name={\noexpand\acronymfont{\the\glsshorttok}},%
4018       sort={\the\glsshorttok},%
4019       text={\the\glsshorttok},%
4020       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4021       short={\the\glsshorttok},%
4022       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4023       long={\the\glslongtok},%
4024       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4025       description={\the\glslongtok},%
4026       descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4027       \the\glskeylisttok
4028     }%
4029   }%
4030   \@do@newglossaryentry
4031 }

```

otnoteAcronymStyle If footnote package option is specified, set the first use to append the long form (stored in description) as a footnote. Use the description key to store the long form.

```

4032 \newcommand*{\SetFootnoteAcronymStyle}{%
4033   \renewcommand{\newacronym}[4][\]{%
4034     \ifx\@glsacronymlists\@empty
4035       \def\@glo@type{\acronymtype}%
4036       \setkeys{glossentry}{##1}%
4037       \DeclareAcronymList{\@glo@type}%
4038       \SetFootnoteAcronymDisplayStyle{\@glo@type}%
4039     \fi
4040     \glskeylisttok{##1}%
4041     \glslabeltok{##2}%
4042     \glsshorttok{##3}%
4043     \glslongtok{##4}%
4044     \newacronymhook
4045     \FootnoteNewAcronymDef
4046   }%

```

Set display

```

4047   \@for\@gls@type:=\@glsacronymlists\do{%
4048     \SetFootnoteAcronymDisplayStyle{\@gls@type}%
4049   }%

```

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.

```

4050 \ifglsmallcaps
4051   \renewcommand*{\acronymfont}[1]{\textsc{##1}}%
4052   \renewcommand*{\acrpluralsuffix}{%
4053     \textup{\glspluralsuffix}}%
4054 \else
4055   \ifglsmaller
4056     \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%
4057   \fi
4058 \fi

  Check for option clash

4059 \ifglacrdua
4060   \PackageError{glossaries}{Option clash: ‘footnote’ and ‘dua’
4061     can’t both be set}{}%
4062 \fi
4063 }%

```

AcronymDisplayStyle Sets the acronym display style for given glossary where neither footnote nor description is required, but smallcaps or smaller specified.

```

4064 \newcommand*{\SetSmallAcronymDisplayStyle}[1]{%
4065   \defglfirstdisplayfirst[#1]{##1##4 (\firstacronymfont{##3})}
4066   \defglfirstdisplay[#1]{\acronymfont{##1}##4}%
4067 }

```

\SmallNewAcronymDef

```

4068 \newcommand*{\SmallNewAcronymDef}{%
4069   \edef\@do@newglossaryentry{%
4070     \noexpand\newglossaryentry{\the\glslabeltok}%
4071     {%
4072       type=\acronymtype,%
4073       name={\noexpand\acronymfont{\the\glsshorttok}},%
4074       sort={\the\glsshorttok},%
4075       text={\noexpand\@glo@symbol},%
4076       %plural={\noexpand\@glo@symbolplural},%
4077       plural={\noexpand\@glo@shortpl},%
4078       first={\the\glslongtok},%
4079       %firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4080       firstplural={\noexpand\@glo@longpl},%
4081       short={\the\glsshorttok},%
4082       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4083       long={\the\glslongtok},%
4084       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4085       description={\noexpand\@glo@first},%
4086       descriptionplural={\noexpand\@glo@firstplural},%
4087       symbol={\the\glsshorttok},%
4088       %symbolplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4089       symbolplural={\noexpand\@glo@shortpl},%
4090       \the\glskeylisttok
4091     }%

```



```

4092 }%
4093 \@do@newglossaryentry
4094 }

```

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.

```

4095 \newcommand*{\SetSmallAcronymStyle}{%
4096   \renewcommand{\newacronym}[4][\]{%
4097     \ifx\@glsacronymlists\@empty
4098       \def\@glo@type{\acronymtype}%
4099       \setkeys{glossentry}{##1}%
4100       \DeclareAcronymList{\@glo@type}%
4101       \SetSmallAcronymDisplayStyle{\@glo@type}%
4102     \fi
4103     \glskeylisttok{##1}%
4104     \glslabeltok{##2}%
4105     \glsshorttok{##3}%
4106     \glslongtok{##4}%
4107     \newacronymhook
4108     \SmallNewAcronymDef
4109   }%

```

Change the display since first only contains long form.

```

4110 \@for\@gls@type:=\@glsacronymlists\do{%
4111   \SetSmallAcronymDisplayStyle{\@gls@type}%
4112 }%

```

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.

```

4113 \ifglsacrsmallcaps
4114   \renewcommand*{\acronymfont}[1]{\textsc{##1}}
4115   \renewcommand*{\acrpluralsuffix}{%
4116     \textup{\glspluralsuffix}}%
4117 \else
4118   \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}
4119 \fi

```

check for option clash

```

4120 \ifglsacrdua
4121   \ifglsacrsmallcaps
4122     \PackageError{glossaries}{Option clash: 'smallcaps' and 'dua'
4123       can't both be set}{}%
4124   \else
4125     \PackageError{glossaries}{Option clash: 'smaller' and 'dua'
4126       can't both be set}{}%
4127   \fi
4128 \fi
4129 }%

```

`\SetDUADisplayStyle` Sets the acronym display style for given glossary with dua setting.

```

4130 \newcommand*\SetDUADisplayStyle}[1]{%
4131   \def\glsdisplay[#1]{##1##4}%
4132   \def\glsdisplayfirst[#1]{##1##4}%
4133 }

```

`\DUANewAcronymDef`

```

4134 \newcommand*\DUANewAcronymDef{%
4135   \edef\@do@newglossaryentry{%
4136     \noexpand\newglossaryentry{\the\glslabeltok}%
4137     {%
4138       type=\acronymtype,%
4139       name={\the\glsshorttok},%
4140       text={\the\glslongtok},%
4141       plural={\the\glslongtok\noexpand\acrpluralsuffix},%
4142       short={\the\glsshorttok},%
4143       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4144       long={\the\glslongtok},%
4145       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4146       description={\the\glslongtok},%
4147       symbol={\the\glsshorttok},%
4148       symbolplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4149       \the\glskeylisttok
4150     }%
4151   }%
4152   \@do@newglossaryentry
4153 }

```

`\SetDUASyle` Always expand acronyms.

```

4154 \newcommand*\SetDUASyle{%
4155   \renewcommand{\newacronym}[4][\]{%
4156     \ifx\@glsacronymlists\@empty
4157       \def\@glo@type{\acronymtype}%
4158       \setkeys{glossentry}{##1}%
4159       \DeclareAcronymList{\@glo@type}%
4160       \SetDUADisplayStyle{\@glo@type}%
4161     \fi
4162     \glskeylisttok{##1}%
4163     \glslabeltok{##2}%
4164     \glsshorttok{##3}%
4165     \glslongtok{##4}%
4166     \newacronymhook
4167     \DUANewAcronymDef
4168   }%

   Set the display
4169   \@for\@gls@type:=\@glsacronymlists\do{%
4170     \SetDUADisplayStyle{\@gls@type}%
4171   }%
4172 }

```

\SetAcronymStyle

```
4173 \newcommand*{\SetAcronymStyle}{%
4174   \SetDefaultAcronymStyle
4175   \ifglssacrdescription
4176     \ifglssacrfootnote
4177       \SetDescriptionFootnoteAcronymStyle
4178     \else
4179       \ifglssacrdua
4180         \SetDescriptionDUAAcronymStyle
4181       \else
4182         \SetDescriptionAcronymStyle
4183       \fi
4184     \fi
4185   \else
4186     \ifglssacrfootnote
4187       \SetFootnoteAcronymStyle
4188     \else
4189       \ifthenelse{\boolean{glssacrsmallsac}s}{OR
4190         \boolean{glssacrsmallsac}}{%
4191         {%
4192           \SetSmallAcronymStyle
4193         }%
4194         {%
4195           \ifglssacrdua
4196             \SetDUASStyle
4197           \fi
4198         }%
4199       \fi
4200     \fi
4201 }
```

Set the acronym style according to the package options

4202 \SetAcronymStyle

Allow user to define their own custom acronyms. (For compatibility with versions before v3.0, the short form is stored in the user1 key, the plural short form is stored in the user2 key, the long form is stored in the user3 key and the plural long form is stored in the user4 key.) Defaults to displaying only the acronym with the long form as the description.

tCustomDisplayStyle Sets the acronym display style.

```
4203 \newcommand*{\SetCustomDisplayStyle}[1]{%
4204   \defglssdisplay[#1]{##1##4}%
4205   \defglssdisplayfirst[#1]{##1##4}%
4206 }
```

CustomAcronymFields

```
4207 \newcommand*{\CustomAcronymFields}{%
4208   name={\the\glssshorttok},%
```

```

4209 description={\the\glslongtok},%
4210 first={\noexpand\acrfullformat{\the\glslongtok}{\the\glsshorttok}},%
4211 firstplural={\noexpand\acrfullformat
4212   {\the\glslongtok\noexpand\acrpluralsuffix}{\the\glsshorttok}}%
4213 text={\the\glsshorttok},%
4214 plural={\the\glsshorttok\noexpand\acrpluralsuffix}%
4215 }

```

CustomNewAcronymDef

```

4216 \newcommand*{\CustomNewAcronymDef}{%
4217   \protected@edef\@do@newglossaryentry{%
4218     \noexpand\newglossaryentry{\the\glslabeltok}%
4219     {%
4220       type=\acronymtype,%
4221       short={\the\glsshorttok},%
4222       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4223       long={\the\glslongtok},%
4224       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4225       user1={\the\glsshorttok},%
4226       user2={\the\glsshorttok\noexpand\acrpluralsuffix},%
4227       user3={\the\glslongtok},%
4228       user4={\the\glslongtok\noexpand\acrpluralsuffix},%
4229       \CustomAcronymFields,%
4230       \the\glskeylisttok
4231     }%
4232   }%
4233   \@do@newglossaryentry
4234 }

```

\SetCustomStyle

```

4235 \newcommand*{\SetCustomStyle}{%
4236   \renewcommand{\newacronym}[4][\]{%
4237     \ifx\@glsacronymlists\@empty
4238       \def\@glo@type{\acronymtype}%
4239       \setkeys{glossentry}{##1}%
4240       \DeclareAcronymList{\@glo@type}%
4241       \SetCustomDisplayStyle{\@glo@type}%
4242     \fi
4243     \glskeylisttok{##1}%
4244     \glslabeltok{##2}%
4245     \glsshorttok{##3}%
4246     \glslongtok{##4}%
4247     \newacronymhook
4248     \CustomNewAcronymDef
4249   }%

```

Set the display

```

4250   \@for\@gls@type:=\@glsacronymlists\do{%
4251     \SetCustomDisplayStyle{\@gls@type}%
4252   }%

```

4253 }

fineAcronymSynonyms

4254 \newcommand*{\DefineAcronymSynonyms}{%

Short form

\acs

4255 \let\acs\acrshort

First letter uppercase short form

\Acs

4256 \let\Acs\Acrshort

Plural short form

\acsp

4257 \let\acsp\acrshortpl

First letter uppercase plural short form

\Acsp

4258 \let\Acsp\Acrshortpl

Long form

\acl

4259 \let\acl\aclong

Plural long form

\aclp

4260 \let\aclp\aclongpl

First letter upper case long form

\Acl

4261 \let\Acl\Aclong

First letter upper case plural long form

\Aclp

4262 \let\Aclp\Aclongpl

Full form

\acf

4263 \let\acf\acrfull

Plural full form

```

\acfp
4264 \let\acfp\acrfullpl

First letter upper case full form

\Acf
4265 \let\Acf\Acrfull

First letter upper case plural full form

\Acfp
4266 \let\Acfp\Acrfullpl

Standard form

\ac
4267 \let\ac\gls

First upper case standard form

\Ac
4268 \let\Ac\Gls

Standard plural form

\acp
4269 \let\acp\glspl

Standard first letter upper case plural form

\Acp
4270 \let\Acp\Glspl
4271 }

Define synonyms if required
4272 \ifglsacrshortcuts
4273 \DefineAcronymSynonyms
4274 \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.

```
4275 \RequirePackage{glossary-hypernav}
```

The styles that use list-like environments. These are not loaded if the `nolist` option is used:

```
4276 \@gls@loadlist
```

The styles that use the longtable environment. These are not loaded if the no-long package option is used.

```
4277 \@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.

```
4278 \@gls@loadsuper
```

The tree-like styles. These are not loaded if the notree package option is used.

```
4279 \@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.

```
4280 \ifx\@glossary@default@style\relax
4281 \else
4282   \glossarystyle{\@glossary@default@style}
4283 \fi
```

1.19 Debugging Commands

\showgloparent \showgloparent{<label>}

```
4284 \newcommand*{\showgloparent}[1]{%
4285   \expandafter\show\csname glo@#1@parent\endcsname
4286 }
```

\showglolevel \showglolevel{<label>}

```
4287 \newcommand*{\showglolevel}[1]{%
4288   \expandafter\show\csname glo@#1@level\endcsname
4289 }
```

\showglotext \showglotext{<label>}

```
4290 \newcommand*{\showglotext}[1]{%
4291   \expandafter\show\csname glo@#1@text\endcsname
4292 }
```

\showgloplural \showgloplural{<label>}

```
4293 \newcommand*{\showgloplural}[1]{%
4294   \expandafter\show\csname glo@#1@plural\endcsname
4295 }
```

\showglofirst \showglofirst{<label>}

```
4296 \newcommand*{\showglofirst}[1]{%
4297   \expandafter\show\csname glo@#1@first\endcsname
4298 }
```

\showglofirstpl \showglofirstpl{<label>}

```
4299 \newcommand*{\showglofirstpl}[1]{%
4300   \expandafter\show\csname glo@#1@firstpl\endcsname
4301 }
```

\showglotype \showglotype{<label>}

```
4302 \newcommand*{\showglotype}[1]{%
4303   \expandafter\show\csname glo@#1@type\endcsname
4304 }
```

\showglocounter \showglocounter{<label>}

```
4305 \newcommand*{\showglocounter}[1]{%
4306   \expandafter\show\csname glo@#1@counter\endcsname
4307 }
```

\showglouser \showglouser{<label>}

```
4308 \newcommand*{\showglouser}[1]{%
4309   \expandafter\show\csname glo@#1@useri\endcsname
4310 }
```

\showglouserii \showglouserii{<label>}

```
4311 \newcommand*{\showglouserii}[1]{%
4312   \expandafter\show\csname glo@#1@userii\endcsname
4313 }
```

\showglouseriii \showglouseriii{<label>}

```
4314 \newcommand*{\showglouseriii}[1]{%
4315   \expandafter\show\csname glo@#1@useriii\endcsname
4316 }
```


\showglouseriv \showglouseriv{<label>}

```
4317 \newcommand*{\showglouseriv}[1]{%
4318   \expandafter\show\csname glo@#1@useriv\endcsname
4319 }
```

\showglouserv \showglouserv{<label>}

```
4320 \newcommand*{\showglouserv}[1]{%
4321   \expandafter\show\csname glo@#1@userv\endcsname
4322 }
```

\showglouservi \showglouservi{<label>}

```
4323 \newcommand*{\showglouservi}[1]{%
4324   \expandafter\show\csname glo@#1@uservi\endcsname
4325 }
```

\showgloname \showgloname{<label>}

```
4326 \newcommand*{\showgloname}[1]{%
4327   \expandafter\show\csname glo@#1@name\endcsname
4328 }
```

\showglodesc \showglodesc{<label>}

```
4329 \newcommand*{\showglodesc}[1]{%
4330   \expandafter\show\csname glo@#1@desc\endcsname
4331 }
```

\showglodescplural \showglodescplural{<label>}

```
4332 \newcommand*{\showglodescplural}[1]{%
4333   \expandafter\show\csname glo@#1@descplural\endcsname
4334 }
```

\showglosort \showglosort{<label>}

```
4335 \newcommand*{\showglosort}[1]{%
4336   \expandafter\show\csname glo@#1@sort\endcsname
4337 }
```

`\showglosymbol` `\showglosymbol{\langle label \rangle}`

```
4338 \newcommand*{\showglosymbol}[1]{%
4339   \expandafter\show\csname glo@#1@symbol\endcsname
4340 }
```

`\showglosymbolplural` `\showglosymbolplural{\langle label \rangle}`

```
4341 \newcommand*{\showglosymbolplural}[1]{%
4342   \expandafter\show\csname glo@#1@symbolplural\endcsname
4343 }
```

`\showgloindex` `\showgloindex{\langle label \rangle}`

```
4344 \newcommand*{\showgloindex}[1]{%
4345   \expandafter\show\csname glo@#1@index\endcsname
4346 }
```

`\showgloflag` `\showgloflag{\langle label \rangle}`

```
4347 \newcommand*{\showgloflag}[1]{%
4348   \expandafter\show\csname ifglo@#1@flag\endcsname
4349 }
```

`\showacronymlists` `\showacronymlists`

Show list of glossaries that have been flagged as a list of acronyms.

```
4350 \newcommand*{\showacronymlists}{%
4351   \show\@glsacronymlists
4352 }
```

`\showglossaries` `\showglossaries`

Show list of defined glossaries.

```
4353 \newcommand*{\showglossaries}{%
4354   \show\@glo@types
4355 }
```

`\showglossaryin` `\showglossaryin{\langle glossary-label \rangle}`

Show the ‘in’ extension for the given glossary.

```
4356 \newcommand*{\showglossaryin}[1]{%  
4357   \expandafter\show\csname @glotype@#1@in\endcsname  
4358 }
```

`\showglossaryout` `\showglossaryout{<glossary-label>}`

Show the ‘out’ extension for the given glossary.

```
4359 \newcommand*{\showglossaryout}[1]{%  
4360   \expandafter\show\csname @glotype@#1@out\endcsname  
4361 }
```

`\showglossarytitle` `\showglossarytitle{<glossary-label>}`

Show the title for the given glossary.

```
4362 \newcommand*{\showglossarytitle}[1]{%  
4363   \expandafter\show\csname @glotype@#1@title\endcsname  
4364 }
```

`\showglossarycounter` `\showglossarycounter{<glossary-label>}`

Show the counter for the given glossary.

```
4365 \newcommand*{\showglossarycounter}[1]{%  
4366   \expandafter\show\csname @glotype@#1@counter\endcsname  
4367 }
```

`\showglossaryentries` `\showglossaryentries{<glossary-label>}`

Show the list of entry labels for the given glossary.

```
4368 \newcommand*{\showglossaryentries}[1]{%  
4369   \expandafter\show\csname glolist@#1\endcsname  
4370 }
```

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.

```
4371 \csname ifglscpatible-2.07\endcsname
4372 \RequirePackage{glossaries-compatible-207}
4373 \fi
```

2 Mfirstuc Documented Code

```
4374 \NeedsTeXFormat{LaTeX2e}
4375 \ProvidesPackage{mfirstuc}[2012/05/21 v1.06 (NLCT)]
```

Requires etoolbox:

```
4376 \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`.

```
4377 \newif\if@glscs
4378 \newtoks\@glsmfirst
4379 \newtoks\@glsmrest
4380 \def\makefirstuc#1{%
4381   \def\gls@argi{#1}%
4382   \ifx\gls@argi\@empty
      If the argument is empty, do nothing.
4383   \else
4384     \def\@gls@tmp{\ #1}%
4385     \@onelevel@sanitize\@gls@tmp
4386     \expandafter\@gls@checkcs\@gls@tmp\relax\relax
4387     \if@glscs
4388       \@gls@getbody #1{}\@nil
4389       \ifx\@gls@rest\@empty
4390         \glsmakefirstuc{#1}%
4391       \else
4392         \expandafter\@gls@split\@gls@rest\@nil
4393         \ifx\@gls@first\@empty
4394           \glsmakefirstuc{#1}%
4395         \else
4396           \expandafter\@glsmfirst\expandafter{\@gls@first}%
```

```

4397         \expandafter\@glsmrest\expandafter{\@gls@rest}%
4398         \edef\@gls@domfirstuc{\noexpand\@gls@body
4399         {\noexpand\glsmakefirstuc\the\@glsmfirst}%
4400         \the\@glsmrest}%
4401         \@gls@domfirstuc
4402     \fi
4403 \fi
4404 \else
4405     \glsmakefirstuc{#1}%
4406 \fi
4407 \fi
4408 }

```

Put first argument in \@gls@first and second argument in \@gls@rest:

```

4409 \def\@gls@split#1#2\@nil{%
4410 \def\@gls@first{#1}\def\@gls@rest{#2}%
4411 }

4412 \def\@gls@checkcs#1 #2#3\relax{%
4413 \def\@gls@argi{#1}\def\@gls@argii{#2}%
4414 \ifx\@gls@argi\@gls@argii
4415     \@glscstrue
4416 \else
4417     \@glscsfalse
4418 \fi
4419 }

```

Make first thing upper case:

```

4420 \def\@gls@makefirstuc#1{\MakeUppercase #1}

```

\glsmakefirstuc Provide a user command to make it easier to customise.

```

4421 \newcommand*\glsmakefirstuc[1]{\@gls@makefirstuc{#1}}

```

Get the first grouped argument and stores in \@gls@body.

```

4422 \def\@gls@getbody#1#\def\@gls@body{#1}\@gls@gobbletonil}

```

Scoop up everything to \@nil and store in \@gls@rest:

```

4423 \def\@gls@gobbletonil#1\@nil{\def\@gls@rest{#1}}

```

\xmakefirstuc Expand argument once before applying \makefirstuc (added v1.01).

```

4424 \newcommand*\xmakefirstuc[1]{%
4425 \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).

```

4426 \newcommand*\capitalisewords[1]{%
4427 \def\gls@add@space{%
4428 \mfu\capitalisewords#1 \@nil\mfu@endcap
4429 %\gls@add@space\makefirstuc{##1}\def\gls@add@space{ }%
4430 }

```

```

4431 \def\mfu@capitalisewords#1 #2\mfu@endcap{%
4432   \def\mfu@cap@first{#1}%
4433   \def\mfu@cap@second{#2}%
4434   \gls@add@space
4435   \makefirstuc{#1}%
4436   \def\gls@add@space{ }%
4437   \ifx\mfu@cap@second\@nnil
4438     \let\next\mfu@cap\mfu@noop
4439   \else
4440     \let\next\mfu@cap\mfu@capitalisewords
4441   \fi
4442   \next\mfu@cap#2\mfu@endcap
4443 }
4444 \def\mfu@noop#1\mfu@endcap{}

```

`\xcapitalisewords` Short-cut command:

```

4445 \newcommand*{\xcapitalisewords}[1]{%
4446   \expandafter\capitalisewords\expandafter{#1}%
4447 }

```

3 Glossary Styles

3.1 Glossary hyper-navigation definitions (glossary-hypernav package)

Package Definition:

```

4448 \ProvidesPackage{glossary-hypernav}[2007/07/04 v1.01 (NLCT)]

```

The commands defined in this package are provided to help navigate around the groups within a glossary (see [subsection 1.15.](#)) `\printglossary` (and `\printglossaries`) set `\@glo@type` to the label of the current glossary. This is used to create a unique hypertext in the event of multiple glossaries.

`\glsnavhyperlink[⟨type⟩]{⟨label⟩}{⟨text⟩}`

This command makes `⟨text⟩` a hyperlink to the glossary group whose label is given by `⟨label⟩` for the glossary given by `⟨type⟩`.

`\glsnavhyperlink`

```

4449 \newcommand*{\glsnavhyperlink}[3][\@glo@type]{%
4450   \edef\gls@grplabel{#2}\protected@edef\@gls@grptitle{#3}%
4451   \@glslink{glsn:#1@#2}{#3}}

```

`\glsnavhypertarget[⟨type⟩]{⟨label⟩}{⟨text⟩}`

This command makes `⟨text⟩` a hypertext for the glossary group whose label is given by `⟨label⟩` in the glossary given by `⟨type⟩`. If `⟨type⟩` is omitted, `\@glo@type` is used which is set by `\printglossary` to the current glossary label.

`\glsnavhypertarget`

```
4452 \newcommand*{\glsnavhypertarget}[3][\@glo@type]{%
```

Add this group to the aux file for re-run check.

```
4453 \protected@write\@auxout{}\string\@gls@hypergroup{#1}{#2}}%
```

Add the target.

```
4454 \@glstarget{glsn:#1@#2}{#3}%
```

Check list of know groups to determine if a re-run is required.

```
4455 \expandafter\let
```

```
4456 \expandafter\@gls@list\csname @gls@hypergroup@list@#1\endcsname
```

Iterate through list and terminate loop if this group is found.

```
4457 \@for\@gls@elem:=\@gls@list\do{%
```

```
4458 \ifthenelse{\equal{\@gls@elem}{#2}}{\@endfortrue}{}}%
```

Check if list terminated prematurely.

```
4459 \if@endfor
```

```
4460 \else
```

This group was not included in the list, so issue a warning.

```
4461 \GlossariesWarningNoLine{Navigation panel
```

```
4462 for glossary type ‘#1’^^Jmissing group ‘#2’}%
```

```
4463 \gdef\gls@hypergroup@rerun{%
```

```
4464 \GlossariesWarningNoLine{Navigation panel
```

```
4465 has changed. Rerun LaTeX}}%
```

```
4466 \fi
```

```
4467 }
```

`\gls@hypergroup@rerun` Give a warning at the end if re-run required

```
4468 \let\gls@hypergroup@rerun\relax
```

```
4469 \AtEndDocument{\gls@hypergroup@rerun}
```

`\@gls@hypergroup` This adds to (or creates) the command `\@gls@hypergroup@list@<glossary type>` which lists all groups for a given glossary, so that the navigation bar only contains those groups that are present. However it requires at least 2 runs to ensure the information is up-to-date.

```
4470 \newcommand*{\@gls@hypergroup}[2]{%
```

```
4471 \@ifundefined{\@gls@hypergroup@list@#1}{%
```

```
4472 \expandafter\xdef\csname @gls@hypergroup@list@#1\endcsname{#2}}%
```

```
4473 }{%
```

```
4474 \expandafter\let\expandafter\@gls@tmp
```

```
4475 \csname @gls@hypergroup@list@#1\endcsname
```

```
4476 \expandafter\xdef\csname @gls@hypergroup@list@#1\endcsname{%
```

```
4477 \@gls@tmp,#2}}%
```

```
4478 }%
```

```
4479 }
```

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`

```
4480 \newcommand*\glsnavigation{%
4481 \def\@gls@between{%
4482 \@ifundefined{@gls@hypergrouplist@\@glo@type}{%
4483   \def\@gls@list{%
4484 }{%
4485   \expandafter\let\expandafter\@gls@list
4486     \csname @gls@hypergrouplist@\@glo@type\endcsname
4487 }%
4488 \@for\@gls@tmp:=\@gls@list\do{%
4489   \@gls@between
4490   \glsnavhyperlink{\@gls@tmp}{\glsgetgrouptitle{\@gls@tmp}}%
4491   \let\@gls@between\glshypernavsep%
4492 }%
4493 }
```

`\glshypernavsep` Separator for the hyper navigation bar.

```
4494 \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`

```
4495 \newcommand*\glssymbolnav{%
4496 \glsnavhyperlink{glssymbols}{\glsgetgrouptitle{glssymbols}}%
4497 \glshypernavsep
4498 \glsnavhyperlink{glsnumbers}{\glsgetgrouptitle{glsnumbers}}%
4499 \glshypernavsep
4500 }
```

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.

```
4501 \ProvidesPackage{glossary-inline}[2012/05/21 v1.0 (NLCT)]
```

`inline` Define the inline style.

```
4502 \newglossarystyle{inline}{%
```


Start of glossary sets up first empty separator between entries. (This is then changed by `\glossaryentryfield`)

```
4503 \renewenvironment{theglossary}%
4504 {%
4505     \def\gls@inlinesep{}%
4506     \def\gls@inlinesubsep{}%
4507 }%
4508 {\glspostinline}%
```

No header:

```
4509 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
4510 \renewcommand*{\glsgroupheading}[1]{}%
```

Just display separator followed by name and description:

```
4511 \renewcommand*{\glossaryentryfield}[5]{%
4512     \gls@inlinesep
4513     \glsentryitem{##1}\glstarget{##1}{##2}%
4514     \def\glo@desc{##3}%
4515     \def\@no@post@desc{\nopostdesc}%
4516     \ifx\glo@desc\@no@post@desc
4517     \else
4518         \ifstrempy{##3}{ }\space##3}%
4519     \fi
4520     \ifglshaschildren{##1}%
4521     {%
4522         \glsresetsubentrycounter
4523         \glsinlineparentchildseparator
4524         \def\gls@inlinesubsep{}%
4525     }%
4526     {}%
4527     \def\gls@inlinesep{\glsinlineseparator}%
4528 }%
```

Sub-entries display description but no name:

```
4529 \renewcommand*{\glossarysubentryfield}[6]{%
4530     \gls@inlinesubsep%
4531     \glstarget{##2}{}%
4532     \glssubentryitem{##2}##4%
4533     \def\gls@inlinesubsep{\glsinlinesubseparator}%
4534 }%
```

Nothing special between groups:

```
4535 \renewcommand*{\glsgroupskip}{}%
4536 }
```

`\glsinlineseparator` Separator to use between entries.

```
4537 \newcommand*{\glsinlineseparator}{;\space}
```

`glsinlinesubseparator` Separator to use between sub-entries.

```
4538 \newcommand*{\glsinlinesubseparator}{,\space}
```

`glsparentchildseparator` Separator to use between parent and children.

```
4539 \newcommand*{\glsinlineparentchildseparator}{:\space}
```

`glspostinline` Terminator for inline glossary.

```
4540 \newcommand*{\glspostinline}{.\space}
```

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.

```
4541 \ProvidesPackage{glossary-list}[2011/03/28 v3.0 (NLCT)]
```

`list` The list glossary style uses the description environment. The group separator `\glsgroupskip` is redefined as `\indexspace` which produces a gap between groups. The glossary heading and the group headings do nothing. Sub-entries immediately follow the main entry without the sub-entry name. This style does not use the entry's symbol. This is used as the default style for the glossaries package.

```
4542 \newglossarystyle{list}{%
```

Use description environment:

```
4543 \renewenvironment{theglossary}{%
```

```
4544 {\begin{description}}{\end{description}}%
```

No header at the start of the environment:

```
4545 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
4546 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries start a new item in the list:

```
4547 \renewcommand*{\glossaryentryfield}[5]{%
```

```
4548 \item[\glsentryitem{##1}\glstarget{##1}{##2}]
```

```
4549 ##3\glspostdescription\space ##5}%
```

Sub-entries continue on the same line:

```
4550 \renewcommand*{\glossarysubentryfield}[6]{%
```

```
4551 \glssubentryitem{##2}%
```

```
4552 \glstarget{##2}{\strut}##4\glspostdescription\space ##6.}%
```

```
4553 % \end{macrocode}
```

```
4554 % Add vertical space between groups:
```

```
4555 % \begin{macrocode}
```

```
4556 \renewcommand*{\glsgroupskip}{\indexspace}%
```

```
4557 }
```

listgroup The listgroup style is like the list style, but the glossary groups have headings.

```
4558 \newglossarystyle{listgroup}{%
```

Base it on the list style:

```
4559 \glossarystyle{list}%
```

Each group has a heading:

```
4560 \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.

```
4561 \newglossarystyle{listhypergroup}{%
```

Base it on the list style:

```
4562 \glossarystyle{list}%
```

Add navigation links at the start of the environment:

```
4563 \renewcommand*{\glossaryheader}{%
```

```
4564 \item[\glsnavigation]}%
```

Each group has a heading with a hypertarget:

```
4565 \renewcommand*{\glsgroupheading}[1]{%
```

```
4566 \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.

```
4567 \newglossarystyle{altlist}{%
```

Base it on the list style:

```
4568 \glossarystyle{list}%
```

Main (level 0) entries start a new item in the list with a line break after the entry name:

```
4569 \renewcommand*{\glossaryentryfield}[5]{%
```

```
4570 \item[\glsentryitem{##1}\glstarget{##1}{##2}]\mbox{}\newline
```

```
4571 ##3\glspostdescription\space ##5}%
```

Sub-entries start a new paragraph:

```
4572 \renewcommand{\glossarysubentryfield}[6]{%
```

```
4573 \par
```

```
4574 \glssubentryitem{##2}%
```

```
4575 \glstarget{##2}{\strut}##4\glspostdescription\space ##6}%
```

```
4576 }
```

altlistgroup The altlistgroup glossary style is like the altlist style, but the glossary groups have headings.

```
4577 \newglossarystyle{altlistgroup}{%
```

Base it on the altlist style:

```
4578 \glossarystyle{altlist}%
```

Each group has a heading:

```
4579 \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.

```
4580 \newglossarystyle{altlisthypergroup}{%
```

Base it on the altlist style:

```
4581 \glossarystyle{altlist}%
```

Add navigation links at the start of the environment:

```
4582 \renewcommand*{\glossaryheader}{%
```

```
4583 \item[\glsnavigation]}%
```

Each group has a heading with a hypertarget:

```
4584 \renewcommand*{\glsgroupheading}[1]{%
```

```
4585 \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.

```
4586 \newglossarystyle{listdotted}{%
```

Base it on the list style:

```
4587 \glossarystyle{list}%
```

Each main (level 0) entry starts a new item:

```
4588 \renewcommand*{\glossaryentryfield}[5]{%
```

```
4589 \item[]\makebox[\glslistdottedwidth][l]{%
```

```
4590 \glentryitem{##1}\glstarget{##1}{##2}%
```

```
4591 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##3}%
```

Sub entries have the same format as main entries:

```
4592 \renewcommand*{\glossarysubentryfield}[6]{%
```

```
4593 \item[]\makebox[\glslistdottedwidth][l]{%
```

```
4594 \glssubentryitem{##2}%
```

```
4595 \glstarget{##2}{##3}%
```

```
4596 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##4}%
```

```
4597 }
```

`\glslistdottedwidth`

```
4598 \newlength\glslistdottedwidth
```

```
4599 \setlength{\glslistdottedwidth}{.5\hsize}
```

sublistdotted This style is similar to the `glostylelistdotted` style, except that the main entries just have the name displayed.

```
4600 \newglossarystyle{sublistdotted}{%
```

Base it on the listdotted style:

```
4601 \glossarystyle{listdotted}%
```

Main (level 0) entries just display the name:

```
4602 \renewcommand*{\glossaryentryfield}[5]{%
4603   \item[\glstentryitem{##1}\glstarget{##1}{##2}]}%
4604 }
```

3.4 Glossary Styles using longtable (the glossary-long package)

The glossary styles defined in the package used the longtable environment in the glossary.

```
4605 \ProvidesPackage{glossary-long}[2011/03/28 v3.0 (NLCT)]
```

Requires the package:

```
4606 \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`.)

```
4607 \@ifundefined{glsdescwidth}{%
4608   \newlength\glsdescwidth
4609   \setlength{\glsdescwidth}{0.6\hsize}
4610 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column.

```
4611 \@ifundefined{glspagelistwidth}{%
4612   \newlength\glspagelistwidth
4613   \setlength{\glspagelistwidth}{0.1\hsize}
4614 }{}
```

`long` The long glossary style command which uses the longtable environment:

```
4615 \newglossarystyle{long}{%
```

Use longtable with two columns:

```
4616   \renewenvironment{theglossary}%
4617     {\begin{longtable}\lp{\glsdescwidth}}%
4618     {\end{longtable}}%
```

Do nothing at the start of the environment:

```
4619   \renewcommand*{\glossaryheader}{}%
```

No heading between groups:

```
4620   \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries displayed in a row:

```
4621   \renewcommand*{\glossaryentryfield}[5]{%
4622     \glstentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\}%
```

Sub entries displayed on the following row without the name:

```
4623 \renewcommand*{\glossarysubentryfield}[6]{%
4624     &
4625     \glssubentryitem{##2}%
4626     \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\}%
```

Blank row between groups:

```
4627 \renewcommand*{\glsgroupskip}{ & \\}%
4628 }
```

longborder The longborder style is like the above, but with horizontal and vertical lines:

```
4629 \newglossarystyle{longborder}{%
```

Base it on the glostylelong style:

```
4630 \glossarystyle{long}%
```

Use longtable with two columns with vertical lines between each column:

```
4631 \renewenvironment{theglossary}{%
4632     \begin{longtable}{|l|p{\glstdescwidth}|}{\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
4633 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
4634 }
```

longheader The longheader style is like the long style but with a header:

```
4635 \newglossarystyle{longheader}{%
```

Base it on the glostylelong style:

```
4636 \glossarystyle{long}%
```

Set the table's header:

```
4637 \renewcommand*{\glossaryheader}{%
4638     \bfseries \entryname & \bfseries \descriptionname\\\endhead}%
4639 }
```

longheaderborder The longheaderborder style is like the long style but with a header and border:

```
4640 \newglossarystyle{longheaderborder}{%
```

Base it on the glostylelongborder style:

```
4641 \glossarystyle{longborder}%
```

Set the table's header and add horizontal line to table's foot:

```
4642 \renewcommand*{\glossaryheader}{%
4643     \hline\bfseries \entryname & \bfseries \descriptionname\\\hline
4644     \endhead
4645     \hline\endfoot}%
4646 }
```

long3col The long3col style is like long but with 3 columns

```
4647 \newglossarystyle{long3col}{%
```

Use a longtable with 3 columns:

```
4648 \renewenvironment{theglossary}%  
4649   {\begin{longtable}{lp{\glstdescwidth}p{\glspagelistwidth}}}%  
4650   {\end{longtable}}%
```

No table header:

```
4651 \renewcommand*{\glossaryheader}{}%
```

No headings between groups:

```
4652 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
4653 \renewcommand*{\glossaryentryfield}[5]{%  
4654   \glstentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\\}%
```

Sub-entries on a separate row (no name, description in second column, page list in third column):

```
4655 \renewcommand*{\glossarysubentryfield}[6]{%  
4656   &  
4657   \glssubentryitem{##2}%  
4658   \glstarget{##2}{\strut}##4 & ##6\\}%
```

Blank row between groups:

```
4659 \renewcommand*{\glsgroupskip}{ & &\\}%  
4660 }
```

long3colborder The long3colborder style is like the long3col style but with a border:

```
4661 \newglossarystyle{long3colborder}{}%
```

Base it on the glostylelong3col style:

```
4662 \glossarystyle{long3col}%
```

Use a longtable with 3 columns with vertical lines around them:

```
4663 \renewenvironment{theglossary}%  
4664   {\begin{longtable}{|lp{\glstdescwidth}|p{\glspagelistwidth}|}}}%  
4665   {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
4666 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%  
4667 }
```

long3colheader The long3colheader style is like long3col but with a header row:

```
4668 \newglossarystyle{long3colheader}{}%
```

Base it on the glostylelong3col style:

```
4669 \glossarystyle{long3col}%
```

Set the table's header:

```
4670 \renewcommand*{\glossaryheader}{}%  
4671   \bfseries\entryname&\bfseries\descriptionname&  
4672   \bfseries\pagelistname\\ \endhead}%  
4673 }
```

long3colheaderborder The long3colheaderborder style is like the above but with a border

```
4674 \newglossarystyle{long3colheaderborder}{%
```

Base it on the glostylelong3colborder style:

```
4675 \glossarystyle{long3colborder}{%
```

Set the table's header and add horizontal line at table's foot:

```
4676 \renewcommand*{\glossaryheader}{%  
4677 \hline  
4678 \bfseries\entryname&\bfseries\descriptionname&  
4679 \bfseries\pagelistname\\ \hline\endhead  
4680 \hline\endfoot}%  
4681 }
```

long4col The long4col style has four columns where the third column contains the value of the associated symbol key.

```
4682 \newglossarystyle{long4col}{%
```

Use a longtable with 4 columns:

```
4683 \renewenvironment{theglossary}{%  
4684 {\begin{longtable}{llll}}%  
4685 {\end{longtable}}%
```

No table header:

```
4686 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
4687 \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):

```
4688 \renewcommand*{\glossaryentryfield}[5]{%  
4689 \glstentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\\}%
```

Sub entries on a single row with no name (description in second column, symbol in third column, page list in last column):

```
4690 \renewcommand*{\glossarysubentryfield}[6]{%  
4691 &  
4692 \glssubentryitem{##2}%  
4693 \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%
```

Blank row between groups:

```
4694 \renewcommand*{\glsgroupskip}{ & & & \\}%  
4695 }
```

long4colheader The long4colheader style is like long4col but with a header row.

```
4696 \newglossarystyle{long4colheader}{%
```

Base it on the glostylelong4col style:

```
4697 \glossarystyle{long4col}{%
```


Table has a header:

```
4698 \renewcommand*{\glossaryheader}{%
4699 \bfseries\entryname&\bfseries\descriptionname&
4700 \bfseries \symbolname&
4701 \bfseries\pagelistname\\\endhead}%
4702 }
```

`long4colborder` The `long4colborder` style is like `long4col` but with a border.

```
4703 \newglossarystyle{long4colborder}{%
  Base it on the glostylelong4col style:
4704 \glossarystyle{long4col}%
  Use a longtable with 4 columns surrounded by vertical lines:
4705 \renewenvironment{theglossary}%
4706 {\begin{longtable}{|l|l|l|l|}}%
4707 {\end{longtable}}%
  Add horizontal lines to the head and foot of the table:
4708 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
4709 }
```

`ong4colheaderborder` The `long4colheaderborder` style is like the above but with a border.

```
4710 \newglossarystyle{long4colheaderborder}{%
  Base it on the glostylelong4col style:
4711 \glossarystyle{long4col}%
  Use a longtable with 4 columns surrounded by vertical lines:
4712 \renewenvironment{theglossary}%
4713 {\begin{longtable}{|l|l|l|l|}}%
4714 {\end{longtable}}%
  Add table header and horizontal line at the table's foot:
4715 \renewcommand*{\glossaryheader}{%
4716 \hline\bfseries\entryname&\bfseries\descriptionname&
4717 \bfseries \symbolname&
4718 \bfseries\pagelistname\\\hline\endhead\hline\endfoot}%
4719 }
```

`altlong4col` The `altlong4col` style is like the `long4col` style but can have multiline descriptions and page lists.

```
4720 \newglossarystyle{altlong4col}{%
  Base it on the glostylelong4col style:
4721 \glossarystyle{long4col}%
  Use a longtable with 4 columns where the second and last columns may have
  multiple lines in each row:
4722 \renewenvironment{theglossary}%
4723 {\begin{longtable}{lp{\glsdescwidth}lp{\glspagelistwidth}}}%
4724 {\end{longtable}}%
4725 }
```

`altlong4colheader` The `altlong4colheader` style is like `altlong4col` but with a header row.

```
4726 \newglossarystyle{altlong4colheader}{%
```

Base it on the `glostylelong4colheader` style:

```
4727 \glossarystyle{long4colheader}%
```

Use a `longtable` with 4 columns where the second and last columns may have multiple lines in each row:

```
4728 \renewenvironment{theglossary}%
```

```
4729 {\begin{longtable}{lp{\glsgdescwidth}lp{\glspagelistwidth}}}%
```

```
4730 {\end{longtable}}%
```

```
4731 }
```

`altlong4colborder` The `altlong4colborder` style is like `altlong4col` but with a border.

```
4732 \newglossarystyle{altlong4colborder}{%
```

Base it on the `glostylelong4colborder` style:

```
4733 \glossarystyle{long4colborder}%
```

Use a `longtable` with 4 columns where the second and last columns may have multiple lines in each row:

```
4734 \renewenvironment{theglossary}%
```

```
4735 {\begin{longtable}{|lp{\glsgdescwidth}|lp{\glspagelistwidth}|}}%
```

```
4736 {\end{longtable}}%
```

```
4737 }
```

`ong4colheaderborder` The `altlong4colheaderborder` style is like the above but with a header as well as a border.

```
4738 \newglossarystyle{altlong4colheaderborder}{%
```

Base it on the `glostylelong4colheaderborder` style:

```
4739 \glossarystyle{long4colheaderborder}%
```

Use a `longtable` with 4 columns where the second and last columns may have multiple lines in each row:

```
4740 \renewenvironment{theglossary}%
```

```
4741 {\begin{longtable}{|lp{\glsgdescwidth}|lp{\glspagelistwidth}|}}%
```

```
4742 {\end{longtable}}%
```

```
4743 }
```

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.

```
4744 \ProvidesPackage{glossary-longragged}[2011/03/28 v3.0 (NLCT)]
```

Requires the package:

```
4745 \RequirePackage{array}
```

Requires the package:

```
4746 \RequirePackage{longtable}
```

`\glsdescwidth` This is a length that governs the width of the description column. This may have already been defined.

```
4747 \@ifundefined{glsdescwidth}{%  
4748   \newlength{glsdescwidth}  
4749   \setlength{glsdescwidth}{0.6\hsize}  
4750 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

```
4751 \@ifundefined{glspagelistwidth}{%  
4752   \newlength{glspagelistwidth}  
4753   \setlength{glspagelistwidth}{0.1\hsize}  
4754 }{}
```

`longragged` The longragged glossary style is like the long but uses ragged right formatting for the description column.

```
4755 \newglossarystyle{longragged}{%
```

Use longtable with two columns:

```
4756   \renewenvironment{theglossary}{%  
4757     {\begin{longtable}{l>{\raggedright}p{glsdescwidth}}}%  
4758     {\end{longtable}}%
```

Do nothing at the start of the environment:

```
4759   \renewcommand*{\glossaryheader}{}%
```

No heading between groups:

```
4760   \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries displayed in a row:

```
4761   \renewcommand*{\glossaryentryfield}[5]{%  
4762     \glstryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%  
4763     \tabularnewline}%
```

Sub entries displayed on the following row without the name:

```
4764   \renewcommand*{\glossarysubentryfield}[6]{%  
4765     &  
4766     \glssubentryitem{##2}%  
4767     \glstarget{##2}{\strut}##4\glspostdescription\space ##6%  
4768     \tabularnewline}%
```

Blank row between groups:

```
4769   \renewcommand*{\glsgroupskip}{ & \tabularnewline}%  
4770 }
```

`longraggedborder` The longraggedborder style is like the above, but with horizontal and vertical lines:

```
4771 \newglossarystyle{longraggedborder}{%
```

Base it on the `glostylelongragged` style:

```
4772 \glossarystyle{longragged}%
```

Use `longtable` with two columns with vertical lines between each column:

```
4773 \renewenvironment{theglossary}{%  
4774 \begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|}%  
4775 {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
4776 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%  
4777 }
```

`longraggedheader` The `longraggedheader` style is like the `longragged` style but with a header:

```
4778 \newglossarystyle{longraggedheader}{%
```

Base it on the `glostylelongragged` style:

```
4779 \glossarystyle{longragged}%
```

Set the table's header:

```
4780 \renewcommand*{\glossaryheader}{%  
4781 \bfseries \entryname & \bfseries \descriptionname  
4782 \tabularnewline\endhead}%  
4783 }
```

`longraggedheaderborder` The `longraggedheaderborder` style is like the `longragged` style but with a header and border:

```
4784 \newglossarystyle{longraggedheaderborder}{%
```

Base it on the `glostylelongraggedborder` style:

```
4785 \glossarystyle{longraggedborder}%
```

Set the table's header and add horizontal line to table's foot:

```
4786 \renewcommand*{\glossaryheader}{%  
4787 \hline\bfseries \entryname & \bfseries \descriptionname  
4788 \tabularnewline\hline  
4789 \endhead  
4790 \hline\endfoot}%  
4791 }
```

`longragged3col` The `longragged3col` style is like `longragged` but with 3 columns

```
4792 \newglossarystyle{longragged3col}{%
```

Use a `longtable` with 3 columns:

```
4793 \renewenvironment{theglossary}{%  
4794 {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}%  
4795 >{\raggedright}p{\glspagelistwidth}}}%  
4796 {\end{longtable}}%
```

No table header:

```
4797 \renewcommand*{\glossaryheader}{}%
```

No headings between groups:

```
4798 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
4799 \renewcommand*{\glossaryentryfield}[5]{%
4800 \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):

```
4801 \renewcommand*{\glossarysubentryfield}[6]{%
4802 &
4803 \glssubentryitem{##2}%
4804 \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%
```

Blank row between groups:

```
4805 \renewcommand*{\glsgroupskip}{ & &\tabularnewline}%
4806 }
```

`longragged3colborder` The `longragged3colborder` style is like the `longragged3col` style but with a border:

```
4807 \newglossarystyle{longragged3colborder}{%
```

Base it on the `glostylelongragged3col` style:

```
4808 \glossarystyle{longragged3col}%
```

Use a longtable with 3 columns with vertical lines around them:

```
4809 \renewenvironment{theglossary}%
4810 {\begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|}%
4811 >{\raggedright}p{\glspagelistwidth}|}%
4812 {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
4813 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
4814 }
```

`longragged3colheader` The `longragged3colheader` style is like `longragged3col` but with a header row:

```
4815 \newglossarystyle{longragged3colheader}{%
```

Base it on the `glostylelongragged3col` style:

```
4816 \glossarystyle{longragged3col}%
```

Set the table's header:

```
4817 \renewcommand*{\glossaryheader}{%
4818 \bfseries\entryname&\bfseries\descriptionname&
4819 \bfseries\pagelistname\tabularnewline\endhead}%
4820 }
```

`longragged3colheaderborder` The `longragged3colheaderborder` style is like the above but with a border

```
4821 \newglossarystyle{longragged3colheaderborder}{%
```

Base it on the `glostylelongragged3colborder` style:

```
4822 \glossarystyle{longragged3colborder}%
```

Set the table's header and add horizontal line at table's foot:

```
4823 \renewcommand*{\glossaryheader}{%
4824 \hline
4825 \bfseries\entryname&\bfseries\descriptionname&
4826 \bfseries\pagelistname\tabularnewline\hline\endhead
4827 \hline\endfoot}%
4828 }
```

`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.

```
4829 \newglossarystyle{altlongragged4col}{%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
4830 \renewenvironment{theglossary}%
4831 {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%
4832 >{\raggedright}p{\glspagelistwidth}}}%
4833 {\end{longtable}}%
```

No table header:

```
4834 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
4835 \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):

```
4836 \renewcommand*{\glossaryentryfield}[5]{%
4837 \glstryitem{##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):

```
4838 \renewcommand*{\glossarysubentryfield}[6]{%
4839 &
4840 \glssubentryitem{##2}%
4841 \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%
```

Blank row between groups:

```
4842 \renewcommand*{\glsgrpskip}{ & & \tabularnewline}%
4843 }
```

`ongragged4colheader` The `altlongragged4colheader` style is like `altlongragged4col` but with a header row.

```
4844 \newglossarystyle{altlongragged4colheader}{%
```

Base it on the `glostylealtlongragged4col` style:

```
4845 \glossarystyle{altlongragged4col}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
4846 \renewenvironment{theglossary}%
4847   {\begin{longtable}{l>{\raggedright}p{\glsgdescwidth}l%
4848     >{\raggedright}p{\glspagelistwidth}}}%
4849   {\end{longtable}}%
```

Table has a header:

```
4850 \renewcommand*{\glossaryheader}{%
4851   \bfseries\entryname&\bfseries\descriptionname&
4852   \bfseries \symbolname&
4853   \bfseries\pagelistname\tabularnewline\endhead}%
4854 }
```

`ongragged4colborder` The `altlongragged4colborder` style is like `altlongragged4col` but with a border.

```
4855 \newglossarystyle{altlongragged4colborder}{%
```

Base it on the `glostylealtlongragged4col` style:

```
4856 \glossarystyle{altlongragged4col}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
4857 \renewenvironment{theglossary}%
4858   {\begin{longtable}{|l|>{\raggedright}p{\glsgdescwidth}|l|%
4859     >{\raggedright}p{\glspagelistwidth}|}%
4860   {\end{longtable}}%
```

Add horizontal lines to the head and foot of the table:

```
4861 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
4862 }
```

`ged4colheaderborder` The `altlongragged4colheaderborder` style is like the above but with a header as well as a border.

```
4863 \newglossarystyle{altlongragged4colheaderborder}{%
```

Base it on the `glostylealtlongragged4col` style:

```
4864 \glossarystyle{altlongragged4col}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
4865 \renewenvironment{theglossary}%
4866   {\begin{longtable}{|l|>{\raggedright}p{\glsgdescwidth}|l|%
4867     >{\raggedright}p{\glspagelistwidth}|}%
4868   {\end{longtable}}%
```

Add table header and horizontal line at the table's foot:

```
4869 \renewcommand*{\glossaryheader}{%
4870   \hline\bfseries\entryname&\bfseries\descriptionname&
4871   \bfseries \symbolname&
4872   \bfseries\pagelistname\tabularnewline\hline\endhead
4873   \hline\endfoot}%
4874 }
```

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.

```
4875 \ProvidesPackage{glossary-mcols}[2012/05/21 v1.0 (NLCT)]
```

Required packages:

```
4876 \RequirePackage{multicol}
```

```
4877 \RequirePackage{glossary-tree}
```

`\glsmcols` Define macro in which to store the number of columns. (Defaults to 2.)

```
4878 \newcommand*\glsmcols{2}
```

`mcolindex` Multi-column index style. Same as the index, but puts the glossary in multiple columns. (Ideally the glossary title should go in the optional argument of `multicols`, but the title isn't part of the glossary style.)

```
4879 \newglossarystyle{mcolindex}{%
4880   \glossarystyle{index}%
4881   \renewenvironment{theglossary}%
4882     {%
4883       \begin{multicols}{2}
4884       \setlength{\parindent}{0pt}%
4885       \setlength{\parskip}{0pt plus 0.3pt}%
4886       \let\item\@idxitem}%
4887   {\end{multicols}}%
4888 }
```

`mcolindexgroup` As `mcolindex` but has headings:

```
4889 \newglossarystyle{mcolindexgroup}{%
4890   \glossarystyle{mcolindex}%
4891   \renewcommand*\glsgroupheading[1]{%
4892     \item\textbf{\glsgroupheading{##1}}\indexspace}%
4893 }
```

`mcolindexhypergroup` The `mcolindexhypergroup` style is like the `mcolindexgroup` style but has hyper navigation.

```
4894 \newglossarystyle{mcolindexhypergroup}{%
```

Base it on the `glostylemcolindex` style:

```
4895   \glossarystyle{mcolindex}%
```

Put navigation links to the groups at the start of the glossary:

```
4896   \renewcommand*\glossaryheader{%
4897     \item\textbf{\glshnavigation}\indexspace}%
```

Add a heading for each group (with a target). The group's title is in bold followed by a vertical gap.

```
4898   \renewcommand*\glsgroupheading[1]{%
4899     \item\textbf{\glshnavhypertarget{##1}}{\glsgroupheading{##1}}}%
4900   \indexspace}%
4901 }
```


mcoltree Multi-column index style. Same as the tree, but puts the glossary in multiple columns.

```
4902 \newglossarystyle{mcoltree}{%
4903   \glossarystyle{tree}%
4904   \renewenvironment{theglossary}%
4905   {%
4906     \begin{multicols}{2}
4907     \setlength{\parindent}{0pt}%
4908     \setlength{\parskip}{0pt plus 0.3pt}%
4909   }%
4910   {\end{multicols}}%
4911 }
```

mcoltreegroup Like the mcoltree style but the glossary groups have headings.

```
4912 \newglossarystyle{mcoltreegroup}{%
  Base it on the glostylemcoltree style:
4913   \glossarystyle{mcoltree}%
  Each group has a heading (in bold) followed by a vertical gap):
4914   \renewcommand{\glsgroupheading}[1]{\par
4915     \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
4916 }
```

mcoltreehypergroup The mcoltreehypergroup style is like the treegroup style, but has a set of links to the groups at the start of the glossary.

```
4917 \newglossarystyle{mcoltreehypergroup}{%
  Base it on the glostylemcoltree style:
4918   \glossarystyle{mcoltree}%
  Put navigation links to the groups at the start of the theglossary environment:
4919   \renewcommand*{\glsgroupheading}[1]{%
4920     \par\noindent\textbf{\glsnavigation}\par\indexspace}%
  Each group has a heading (in bold with a target) followed by a vertical gap):
4921   \renewcommand*{\glsgroupheading}[1]{%
4922     \par\noindent
4923     \textbf{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}\par
4924     \indexspace}%
4925 }
```

mcoltreename Multi-column index style. Same as the treename, but puts the glossary in multiple columns.

```
4926 \newglossarystyle{mcoltreename}{%
4927   \glossarystyle{treename}%
4928   \renewenvironment{theglossary}%
4929   {%
4930     \begin{multicols}{2}
4931     \setlength{\parindent}{0pt}%

```

```

4932     \setlength{\parskip}{0pt plus 0.3pt}%
4933   }%
4934   {\end{multicols}}}%
4935 }

```

mcoltreenonamegroup Like the mcoltreenoname style but the glossary groups have headings.

```

4936 \newglossarystyle{mcoltreenonamegroup}{%
    Base it on the glostylemcoltreenoname style:
4937   \glossarystyle{mcoltreenoname}%
    Give each group a heading:
4938   \renewcommand{\glsgroupheading}[1]{\par
4939     \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
4940 }

```

treenonamehypergroup The mcoltreenonamehypergroup style is like the mcoltreenonamegroup style, but has a set of links to the groups at the start of the glossary.

```

4941 \newglossarystyle{mcoltreenonamehypergroup}{%
    Base it on the glostylemcoltreenoname style:
4942   \glossarystyle{mcoltreenoname}%
    Put navigation links to the groups at the start of the theglossary environment:
4943   \renewcommand*{\glossaryheader}{%
4944     \par\noindent\textbf{\glsnavigation}\par\indexspace}%
    Each group has a heading (in bold with a target) followed by a vertical gap):
4945   \renewcommand*{\glsgroupheading}[1]{%
4946     \par\noindent
4947     \textbf{\glsnavigationhypertarget{##1}{\glsgrouptitle{##1}}}\par
4948     \indexspace}%
4949 }

```

mcolalmtree Multi-column index style. Same as the almtree, but puts the glossary in multiple columns.

```

4950 \newglossarystyle{mcolalmtree}{%
4951   \glossarystyle{almtree}%
4952   \renewenvironment{theglossary}%
4953   {%
4954     \begin{multicols}{2}%
4955     \def\@gls@prevlevel{-1}%
4956     \mbox{}\par
4957   }%
4958   {\par\end{multicols}}%
4959 }

```

mcolalmtreegroup Like the mcolalmtree style but the glossary groups have headings.

```

4960 \newglossarystyle{mcolalmtreegroup}{%

```

Base it on the `glostylemcolalttree` style:

```

4961 \glossarystyle{mcolalttree}%
Give each group a heading.
4962 \renewcommand{\glsgroupheading}[1]{\par
4963 \def\@gls@prevlevel{-1}%
4964 \hangindent0pt\relax
4965 \parindent0pt\relax
4966 \textbf{\glsgetgrouptitle{##1}}\par\indexspace}%
4967 }

```

`colalttreehypergroup` The `mcolalttreehypergroup` style is like the `mcolalttreegroup` style, but has a set of links to the groups at the start of the glossary.

```

4968 \newglossarystyle{mcolalttreehypergroup}{%

```

Base it on the `glostylemcolalttree` style:

```

4969 \glossarystyle{mcolalttree}%
Put the navigation links in the header
4970 \renewcommand*{\glossaryheader}{%
4971 \par
4972 \def\@gls@prevlevel{-1}%
4973 \hangindent0pt\relax
4974 \parindent0pt\relax
4975 \textbf{\glsnavigation}\par\indexspace}%

```

Put a hypertarget at the start of each group

```

4976 \renewcommand*{\glsgroupheading}[1]{%
4977 \par
4978 \def\@gls@prevlevel{-1}%
4979 \hangindent0pt\relax
4980 \parindent0pt\relax
4981 \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
4982 \indexspace}}

```

3.7 Glossary Styles using supertabular environment (glossary-super package)

The glossary styles defined in the package use the `supertabular` environment.

```

4983 \ProvidesPackage{glossary-super}[2011/03/28 v3.0 (NLCT)]

```

Requires the package:

```

4984 \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.

```

4985 \@ifundefined{glsdescwidth}{%
4986 \newlength\glsdescwidth
4987 \setlength{\glsdescwidth}{0.6\hsize}
4988 }{}

```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined if has been loaded.

```
4989 \@ifundefined{glspagelistwidth}{%
4990   \newlength{glspagelistwidth
4991   \setlength{glspagelistwidth}{0.1\hsize}
4992 }{}
```

super The super glossary style uses the supertabular environment (it uses lengths defined in the package.)

```
4993 \newglossarystyle{super}{%
```

Put the glossary in a supertabular environment with two columns and no head or tail:

```
4994   \renewenvironment{theglossary}%
4995     {\tablehead{}\tabletail}%
4996     \begin{supertabular}{lp{glstdescwidth}}%
4997     {\end{supertabular}}%
```

Do nothing at the start of the table:

```
4998   \renewcommand*{glossaryheader}{}%
```

No group headings:

```
4999   \renewcommand*{glsgroupheading}[1]{}%
```

Main (level 0) entries put in a row (name in first column, description and page list in second column):

```
5000   \renewcommand*{glossaryentryfield}[5]{}%
5001   \glstryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\}%
```

Sub entries put in a row (no name, description and page list in second column):

```
5002   \renewcommand*{glossarysubentryfield}[6]{}%
5003   &
5004   \glssubentryitem{##2}%
5005   \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\}%
```

Blank row between groups:

```
5006   \renewcommand*{glsgroupskip}{ & \\}%
5007 }
```

superborder The superborder style is like the above, but with horizontal and vertical lines:

```
5008 \newglossarystyle{superborder}{%
```

Base it on the `glostylesuper` style:

```
5009   \glossarystyle{super}%
```

Put the glossary in a supertabular environment with two columns and a horizontal line in the head and tail:

```
5010   \renewenvironment{theglossary}%
5011     {\tablehead{\hline}\tabletail{\hline}%
5012     \begin{supertabular}{|lp{glstdescwidth}|}%
5013     {\end{supertabular}}%
5014 }
```

superheader The superheader style is like the super style, but with a header:

```
5015 \newglossarystyle{superheader}{%
    Base it on the glostylesuper style:
5016   \glossarystyle{super}%
    Put the glossary in a supertabular environment with two columns, a header and
    no tail:
5017 \renewenvironment{theglossary}%
5018   {\tablehead{\bfseries \entryname & \bfseries \descriptionname\\}%
5019    \tabletail{}}%
5020   \begin{supertabular}{lp{\glsgdescwidth}}}%
5021   {\end{supertabular}}%
5022 }
```

superheaderborder The superheaderborder style is like the super style but with a header and border:

```
5023 \newglossarystyle{superheaderborder}{%
    Base it on the glostylesuper style:
5024   \glossarystyle{super}%
    Put the glossary in a supertabular environment with two columns, a header and
    horizontal lines above and below the table:
5025   \renewenvironment{theglossary}%
5026     {\tablehead{\hline\bfseries \entryname &
5027                  \bfseries \descriptionname\\ \hline}%
5028     \tabletail{\hline}
5029     \begin{supertabular}{|lp{\glsgdescwidth}|}%
5030     {\end{supertabular}}%
5031 }
```

super3col The super3col style is like the super style, but with 3 columns:

```
5032 \newglossarystyle{super3col}{%
    Put the glossary in a supertabular environment with three columns and no head
    or tail:
5033   \renewenvironment{theglossary}%
5034     {\tablehead{} \tabletail{}}%
5035     \begin{supertabular}{lp{\glsgdescwidth}p{\glspagelistwidth}}}%
5036     {\end{supertabular}}%
    Do nothing at the start of the table:
5037   \renewcommand*{\glossaryheader}{}%
    No group headings:
5038   \renewcommand*{\glsgroupheading}[1]{}%
    Main (level 0) entries on a row (name in first column, description in second
    column, page list in last column):
5039   \renewcommand*{\glossaryentryfield}[5]{%
5040     \glstentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\\}%

```

Sub entries on a row (no name, description in second column, page list in last column):

```
5041 \renewcommand*{\glossarysubentryfield}[6]{%
5042     &
5043     \glssubentryitem{##2}%
5044     \glstarget{##2}{\strut}##4 & ##6\\}%
```

Blank row between groups:

```
5045 \renewcommand*{\glsgroupskip}{ & &\\}%
5046 }
```

super3colborder The super3colborder style is like the super3col style, but with a border:

```
5047 \newglossarystyle{super3colborder}{%
    Base it on the glostylesuper3col style:
5048 \glossarystyle{super3col}%
    Put the glossary in a supertabular environment with three columns and a horizontal line in the head and tail:
5049 \renewenvironment{theglossary}%
5050     {\tablehead{\hline}\tabletail{\hline}%
5051     \begin{supertabular}{|l|p{\glsgdescwidth}|p{\glspagelistwidth}|}%
5052     {\end{supertabular}}}%
5053 }
```

super3colheader The super3colheader style is like the super3col style but with a header row:

```
5054 \newglossarystyle{super3colheader}{%
    Base it on the glostylesuper3col style:
5055 \glossarystyle{super3col}%
    Put the glossary in a supertabular environment with three columns, a header and no tail:
5056 \renewenvironment{theglossary}%
5057     {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
5058     \bfseries\pagelistname\\}\tabletail{}}%
5059     \begin{supertabular}{\lp{\glsgdescwidth}p{\glspagelistwidth}}}%
5060     {\end{supertabular}}}%
5061 }
```

super3colheaderborder The super3colheaderborder style is like the super3col style but with a header and border:

```
5062 \newglossarystyle{super3colheaderborder}{%
    Base it on the glostylesuper3colborder style:
5063 \glossarystyle{super3colborder}%
    Put the glossary in a supertabular environment with three columns, a header with horizontal lines and a horizontal line in the tail:
5064 \renewenvironment{theglossary}%
5065     {\tablehead{\hline
```

```

5066      \bfseries\entryname&\bfseries\descriptionname&
5067      \bfseries\pagelistname\\\hline}%
5068      \tabletail{\hline}%
5069      \begin{supertabular}{|l|p{\glsgdescwidth}|p{\glspagelistwidth}|}%
5070      {\end{supertabular}}}%
5071 }

```

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.

```
5072 \newglossarystyle{super4col}{%
```

Put the glossary in a supertabular environment with four columns and no head or tail:

```

5073   \renewenvironment{theglossary}%
5074   {\tablehead{}\tabletail}%
5075   \begin{supertabular}{|l|l|l|l|}%
5076   \end{supertabular}}%

```

Do nothing at the start of the table:

```
5077   \renewcommand*\glossaryheader{}%
```

No group headings:

```
5078   \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:

```

5079   \renewcommand*\glossaryentryfield[5]{%
5080     \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:

```

5081   \renewcommand*\glossarysubentryfield[6]{%
5082     &
5083     \glssubentryitem{##2}%
5084     \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%

```

Blank row between groups:

```

5085   \renewcommand*\glsgroupskip{ & & & \\}%
5086 }

```

super4colheader The **super4colheader** style is like the **super4col** but with a header row.

```
5087 \newglossarystyle{super4colheader}{%
```

Base it on the **glostylesuper4col** style:

```
5088   \glossarystyle{super4col}%
```

Put the glossary in a supertabular environment with four columns, a header and no tail:

```

5089   \renewenvironment{theglossary}%
5090   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
5091     \bfseries\symbolname &

```

```

5092      \bfseries\pagelistname\\}%
5093      \tabletail{}}%
5094      \begin{supertabular}{|l|l|l|l|}%
5095      {\end{supertabular}}}%
5096 }

```

super4colborder The super4colborder style is like the super4col but with a border.

```
5097 \newglossarystyle{super4colborder}{%
```

Base it on the glostylesuper4col style:

```
5098 \glossarystyle{super4col}%
```

Put the glossary in a supertabular environment with four columns and a horizontal line in the head and tail:

```

5099 \renewenvironment{theglossary}%
5100 {\tablehead{\hline}\tabletail{\hline}%
5101 \begin{supertabular}{|l|l|l|l|}%
5102 {\end{supertabular}}}%
5103 }

```

super4colheaderborder The super4colheaderborder style is like the super4col but with a header and border.

```
5104 \newglossarystyle{super4colheaderborder}{%
```

Base it on the glostylesuper4col style:

```
5105 \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:

```

5106 \renewenvironment{theglossary}%
5107 {\tablehead{\hline\bfseries\entryname&\bfseries\descriptionname&
5108 \bfseries\symbolname &
5109 \bfseries\pagelistname\\hline}\tabletail{\hline}%
5110 \begin{supertabular}{|l|l|l|l|}%
5111 {\end{supertabular}}}%
5112 }

```

altsuper4col The altsuper4col glossary style is like super4col but has provision for multiline descriptions.

```
5113 \newglossarystyle{altsuper4col}{%
```

Base it on the glostylesuper4col style:

```
5114 \glossarystyle{super4col}%
```

Put the glossary in a supertabular environment with four columns and no head or tail:

```

5115 \renewenvironment{theglossary}%
5116 {\tablehead{} \tabletail{}}%
5117 \begin{supertabular}{lp{\glsgdescwidth}lp{\glspagelistwidth}}}%
5118 {\end{supertabular}}}%
5119 }

```


altsuper4colheader The altsuper4colheader style is like the altsuper4col but with a header row.

```

5120 \newglossarystyle{altsuper4colheader}{%
    Base it on the glostylesuper4colheader style:
5121   \glossarystyle{super4colheader}%

    Put the glossary in a supertabular environment with four columns, a header and
    no tail:
5122   \renewenvironment{theglossary}%
5123     {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
5124       \bfseries\symbolname &
5125       \bfseries\pagelistname\\}\tabletail{}}%
5126     \begin{supertabular}{lp{\glsgdescwidth}lp{\glspagelistwidth}}}%
5127     {\end{supertabular}}}%
5128 }
```

altsuper4colborder The altsuper4colborder style is like the altsuper4col but with a border.

```

5129 \newglossarystyle{altsuper4colborder}{%
    Base it on the glostylesuper4colborder style:
5130   \glossarystyle{super4colborder}%

    Put the glossary in a supertabular environment with four columns and a hori-
    zontal line in the head and tail:
5131   \renewenvironment{theglossary}%
5132     {\tablehead{\hline}\tabletail{\hline}%
5133     \begin{supertabular}%
5134       {\lllp{\glsgdescwidth}lllp{\glspagelistwidth}l}}%
5135     {\end{supertabular}}}%
5136 }
```

per4colheaderborder The altsuper4colheaderborder style is like the altsuper4col but with a header and border.

```

5137 \newglossarystyle{altsuper4colheaderborder}{%
    Base it on the glostylesuper4colheaderborder style:
5138   \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:
5139   \renewenvironment{theglossary}%
5140     {\tablehead{\hline
5141       \bfseries\entryname &
5142       \bfseries\descriptionname &
5143       \bfseries\symbolname &
5144       \bfseries\pagelistname\\\hline}%
5145     \tabletail{\hline}%
5146     \begin{supertabular}%
5147       {\lllp{\glsgdescwidth}lllp{\glspagelistwidth}l}}%
5148     {\end{supertabular}}}%
5149 }
```

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.

5150 \ProvidesPackage{glossary-superragged}[2011/03/28 v3.0 (NLCT)]

Requires the package:

5151 \RequirePackage{array}

Requires the package:

5152 \RequirePackage{supertabular}

`\glsdescwidth` This is a length that governs the width of the description column. This may already have been defined.

```
5153 \@ifundefined{glsdescwidth}{%
5154   \newlength{glsdescwidth
5155   \setlength{glsdescwidth}{0.6\hspace}
5156 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

```
5157 \@ifundefined{glspagelistwidth}{%
5158   \newlength{glspagelistwidth
5159   \setlength{glspagelistwidth}{0.1\hspace}
5160 }{}
```

`superragged` The superragged glossary style uses the supertabular environment.

5161 \newglossarystyle{superragged}{%

Put the glossary in a supertabular environment with two columns and no head or tail:

```
5162   \renewenvironment{theglossary}%
5163   {\tablehead{}\tabletail}%
5164   \begin{supertabular}{1>\raggedright}p{glsdescwidth}}%
5165   {\end{supertabular}}%
```

Do nothing at the start of the table:

5166 \renewcommand*{\glossaryheader}{}%

No group headings:

5167 \renewcommand*{\glsgroupheading}[1]{}%

Main (level 0) entries put in a row (name in first column, description and page list in second column):

```
5168 \renewcommand*{\glossaryentryfield}[5]{%
5169   \glstryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%
5170   \tabularnewline}%
```

Sub entries put in a row (no name, description and page list in second column):

```
5171 \renewcommand*{\glossarysubentryfield}[6]{%
5172     &
5173     \glssubentryitem{##2}%
5174     \glstarget{##2}{\strut}##4\glspostdescription\space ##6%
5175     \tabularnewline}%
```

Blank row between groups:

```
5176 \renewcommand*{\glsgroupskip}{ & \tabularnewline}%
5177 }
```

superraggedborder The superraggedborder style is like the above, but with horizontal and vertical lines:

```
5178 \newglossarystyle{superraggedborder}{%
```

Base it on the glostylesuperragged style:

```
5179 \glossarystyle{superragged}%
```

Put the glossary in a supertabular environment with two columns and a horizontal line in the head and tail:

```
5180 \renewenvironment{theglossary}%
5181     {\tablehead{\hline}\tabletail{\hline}%
5182     \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}|}}%
5183     {\end{supertabular}}%
5184 }
```

superraggedheader The superraggedheader style is like the super style, but with a header:

```
5185 \newglossarystyle{superraggedheader}{%
```

Base it on the glostylesuperragged style:

```
5186 \glossarystyle{superragged}%
```

Put the glossary in a supertabular environment with two columns, a header and no tail:

```
5187 \renewenvironment{theglossary}%
5188     {\tablehead{\bfseries \entryname & \bfseries \descriptionname
5189     \tabularnewline}%
5190     \tabletail{}}%
5191     \begin{supertabular}{l>{\raggedright}p{\glsgdescwidth}}}%
5192     {\end{supertabular}}%
5193 }
```

rraggedheaderborder The superraggedheaderborder style is like the superragged style but with a header and border:

```
5194 \newglossarystyle{superraggedheaderborder}{%
```

Base it on the glostylesuper style:

```
5195 \glossarystyle{superragged}%
```

Put the glossary in a supertabular environment with two columns, a header and horizontal lines above and below the table:

```

5196 \renewenvironment{theglossary}%
5197   {\tablehead{\hline\bfseries \entryname &
5198     \bfseries \descriptionname\tabularnewline\hline}%
5199   \tabletail{\hline}
5200   \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}|}}%
5201   {\end{supertabular}}%
5202 }

```

superragged3col The superragged3col style is like the superragged style, but with 3 columns:

```

5203 \newglossarystyle{superragged3col}{%

```

Put the glossary in a supertabular environment with three columns and no head or tail:

```

5204 \renewenvironment{theglossary}%
5205   {\tablehead{}\tabletail{}}%
5206   \begin{supertabular}{|l>{\raggedright}p{\glsgdescwidth}%
5207     >{\raggedright}p{\glspagelistwidth}}}%
5208   {\end{supertabular}}%

```

Do nothing at the start of the table:

```

5209 \renewcommand*{\glossaryheader}{}%

```

No group headings:

```

5210 \renewcommand*{\glsgroupheading}[1]{}%

```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```

5211 \renewcommand*{\glossaryentryfield}[5]{%
5212   \glstarget{##1}{\glstarget{##1}{##2} & ##3 & ##5\tabularnewline}%

```

Sub entries on a row (no name, description in second column, page list in last column):

```

5213 \renewcommand*{\glossarysubentryfield}[6]{%
5214   &
5215   \glssubentryitem{##2}%
5216   \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%

```

Blank row between groups:

```

5217 \renewcommand*{\glsgroupskip}{ & &\tabularnewline}%
5218 }

```

superragged3colborder The superragged3colborder style is like the superragged3col style, but with a border:

```

5219 \newglossarystyle{superragged3colborder}{%

```

Base it on the glostylesuperragged3col style:

```

5220 \glossarystyle{superragged3col}%

```

Put the glossary in a supertabular environment with three columns and a horizontal line in the head and tail:

```
5221 \renewenvironment{theglossary}%
5222   {\tablehead{\hline}\tabletail{\hline}%
5223    \begin{supertabular}{|l|>{\raggedright}p{\glsdescwidth}|%
5224     >{\raggedright}p{\glspagelistwidth}|}%
5225    {\end{supertabular}}%
5226 }
```

superragged3colheader The superragged3colheader style is like the superragged3col style but with a header row:

```
5227 \newglossarystyle{superragged3colheader}{%
```

Base it on the glostylesuperragged3col style:

```
5228 \glossarystyle{superragged3col}%
```

Put the glossary in a supertabular environment with three columns, a header and no tail:

```
5229 \renewenvironment{theglossary}%
5230   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
5231    \bfseries\pagelistname\tabularnewline}\tabletail{}}%
5232   \begin{supertabular}{|l>{\raggedright}p{\glsdescwidth}%
5233    >{\raggedright}p{\glspagelistwidth}}%
5234   {\end{supertabular}}%
5235 }
```

superragged3colheaderborder The superragged3colheaderborder style is like the superragged3col style but with a header and border:

```
5236 \newglossarystyle{superragged3colheaderborder}{%
```

Base it on the glostylesuperragged3colborder style:

```
5237 \glossarystyle{superragged3colborder}%
```

Put the glossary in a supertabular environment with three columns, a header with horizontal lines and a horizontal line in the tail:

```
5238 \renewenvironment{theglossary}%
5239   {\tablehead{\hline
5240    \bfseries\entryname&\bfseries\descriptionname&
5241    \bfseries\pagelistname\tabularnewline\hline}%
5242   \tabletail{\hline}%
5243   \begin{supertabular}{|l|>{\raggedright}p{\glsdescwidth}|%
5244    >{\raggedright}p{\glspagelistwidth}|}%
5245   {\end{supertabular}}%
5246 }
```

altsuperragged4col The altsuperragged4col glossary style is like altsuper4col style in the package but uses ragged right formatting in the description and page list columns.

```
5247 \newglossarystyle{altsuperragged4col}{%
```

Put the glossary in a supertabular environment with four columns and no head or tail:

```
5248 \renewenvironment{theglossary}%
5249   {\tablehead{}\tabletail{}}%
5250   \begin{supertabular}{1>{\raggedright}p{\glsgdescwidth}1%
5251     >{\raggedright}p{\glspagelistwidth}}}%
5252   {\end{supertabular}}%
```

Do nothing at the start of the table:

```
5253 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
5254 \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:

```
5255 \renewcommand*{\glossaryentryfield}[5]{%
5256   \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:

```
5257 \renewcommand*{\glossarysubentryfield}[6]{%
5258   &
5259   \glssubentryitem{##2}%
5260   \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%
```

Blank row between groups:

```
5261 \renewcommand*{\glsgrpskip}{ & & \tabularnewline}%
5262 }
```

`\perragged4colheader` The `altsuperragged4colheader` style is like the `altsuperragged4col` style but with a header row.

```
5263 \newglossarystyle{altsuperragged4colheader}{%
```

Base it on the `glostylealtsuperragged4col` style:

```
5264 \glossarystyle{altsuperragged4col}%
```

Put the glossary in a supertabular environment with four columns, a header and no tail:

```
5265 \renewenvironment{theglossary}%
5266   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
5267     \bfseries\symbolname &
5268     \bfseries\pagelistname\tabularnewline}\tabletail{}}%
5269   \begin{supertabular}{1>{\raggedright}p{\glsgdescwidth}1%
5270     >{\raggedright}p{\glspagelistwidth}}}%
5271   {\end{supertabular}}%
5272 }
```

`\perragged4colborder` The `altsuperragged4colborder` style is like the `altsuperragged4col` style but with a border.

```
5273 \newglossarystyle{altsuperragged4colborder}{%
```

Base it on the `glostylealtsuperragged4col` style:

```
5274 \glossarystyle{altsuper4col}%
```

Put the glossary in a `supertabular` environment with four columns and a horizontal line in the head and tail:

```
5275 \renewenvironment{theglossary}%
5276   {\tablehead{\hline}\tabletail{\hline}%
5277   \begin{supertabular}%
5278     {\ll>{\raggedright}p{\glsgdescwidth}\ll%
5279     >{\raggedright}p{\glspagelistwidth}\l}%
5280   {\end{supertabular}}%
5281 }
```

`ged4colheaderborder` The `altsuperragged4colheaderborder` style is like the `altsuperragged4col` style but with a header and border.

```
5282 \newglossarystyle{altsuperragged4colheaderborder}{%
```

Base it on the `glostylealtsuperragged4col` style:

```
5283 \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:

```
5284 \renewenvironment{theglossary}%
5285   {\tablehead{\hline
5286     \bfseries\entryname &
5287     \bfseries\descriptionname &
5288     \bfseries\symbolname &
5289     \bfseries\pagelistname\tabularnewline\hline}%
5290   \tabletail{\hline}%
5291   \begin{supertabular}%
5292     {\ll>{\raggedright}p{\glsgdescwidth}\ll%
5293     >{\raggedright}p{\glspagelistwidth}\l}%
5294   {\end{supertabular}}%
5295 }
```

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.

```
5296 \ProvidesPackage{glossary-tree}[2011/03/28 v3.0 (NLCT)]
```

`index` The index glossary style is similar in style to the way indices are usually typeset using `\item`, `\subitem` and `\subsubitem`. The entry name is set in bold. If an entry has a symbol, it is placed in brackets after the name. Then the description is displayed, followed by the number list. This style allows up to three levels.

```
5297 \newglossarystyle{index}{%
```

Set the paragraph indentation and skip and define `\item` to be the same as that used by `theindex`:

```

5298 \renewenvironment{theglossary}%
5299   {\setlength{\parindent}{0pt}%
5300    \setlength{\parskip}{0pt plus 0.3pt}%
5301    \let\item\@idxitem}%
5302   {}%

```

Do nothing at the start of the environment:

```

5303 \renewcommand*{\glossaryheader}{}%

```

No group headers:

```

5304 \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.

```

5305 \renewcommand*{\glossaryentryfield}[5]{%
5306 \item\glsentryitem{##1}\textbf{\glstarget{##1}{##2}}%
5307 \ifx\relax##4\relax
5308 \else
5309   \space{##4}%
5310 \fi
5311 \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.

```

5312 \renewcommand*{\glossarysubentryfield}[6]{%
5313   \ifcase##1\relax
5314     % level 0
5315     \item
5316   \or
5317     % level 1
5318     \subitem
5319     \glssubentryitem{##2}%
5320   \else
5321     % all other levels
5322     \subsubitem
5323   \fi
5324   \textbf{\glstarget{##2}{##3}}%
5325   \ifx\relax##5\relax
5326   \else
5327     \space{##5}%
5328   \fi
5329   \space##4\glspostdescription\space ##6}%

```

Vertical gap between groups is the same as that used by indices:

```

5330 \renewcommand*{\glsgroupskip}{\indexspace}

```

`indexgroup` The `indexgroup` style is like the `index` style but has headings.

```

5331 \newglossarystyle{indexgroup}{%

```


Base it on the `glostyleindex` style:

```
5332 \glossarystyle{index}%
```

Add a heading for each group. This puts the group's title in bold followed by a vertical gap.

```
5333 \renewcommand*{\glsgroupheading}[1]{%
5334   \item\textbf{\glsgetgrouptitle{##1}}\indexspace}%
5335 }
```

`indexhypergroup` The `indexhypergroup` style is like the `indexgroup` style but has hyper navigation.

```
5336 \newglossarystyle{indexhypergroup}{%
```

Base it on the `glostyleindex` style:

```
5337 \glossarystyle{index}%
```

Put navigation links to the groups at the start of the glossary:

```
5338 \renewcommand*{\glossaryheader}{%
5339   \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.

```
5340 \renewcommand*{\glsgroupheading}[1]{%
5341   \item\textbf{\glsnavhypertarget{##1}}{\glsgetgrouptitle{##1}}}%
5342   \indexspace}%
5343 }
```

`tree` The `tree` glossary style is similar in style to the `index` style, but can have arbitrary levels.

```
5344 \newglossarystyle{tree}{%
```

Set the paragraph indentation and skip:

```
5345 \renewenvironment{theglossary}%
5346   {\setlength{\parindent}{0pt}%
5347    \setlength{\parskip}{0pt plus 0.3pt}}%
5348   {}%
```

Do nothing at the start of the `theglossary` environment:

```
5349 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
5350 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries: name in bold, followed by symbol in brackets (if it exists), the description and the page list:

```
5351 \renewcommand{\glossaryentryfield}[5]{%
5352   \hangindent0pt\relax
5353   \parindent0pt\relax
5354   \glsentryitem{##1}\textbf{\glstarget{##1}{##2}}%
5355   \ifx\relax##4\relax
5356   \else
5357     \space{##4}%
5358   \fi
5359   \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.

```

5360 \renewcommand{\glossarysubentryfield}[6]{%
5361     \hangindent##1\glstreeindent\relax
5362     \parindent##1\glstreeindent\relax
5363     \ifnum##1=1\relax
5364         \glssubentryitem{##2}%
5365     \fi
5366     \textbf{\glstarget{##2}{##3}}%
5367     \ifx\relax##5\relax
5368     \else
5369         \space{##5}%
5370     \fi
5371     \space##4\glspostdescription\space ##6\par}%

```

Vertical gap between groups is the same as that used by indices:

```

5372 \renewcommand*{\glsgroupskip}{\indexspace}

```

treegroup Like the tree style but the glossary groups have headings.

```

5373 \newglossarystyle{treegroup}{%

```

Base it on the `glostyletree` style:

```

5374 \glossarystyle{tree}%

```

Each group has a heading (in bold) followed by a vertical gap):

```

5375 \renewcommand{\glsgroupheading}[1]{\par
5376     \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
5377 }

```

treehypergroup The `treehypergroup` style is like the `treegroup` style, but has a set of links to the groups at the start of the glossary.

```

5378 \newglossarystyle{treehypergroup}{%

```

Base it on the `glostyletree` style:

```

5379 \glossarystyle{tree}%

```

Put navigation links to the groups at the start of the `theglossary` environment:

```

5380 \renewcommand*{\glossaryheader}{%
5381     \par\noindent\textbf{\glsnavigation}\par\indexspace}%

```

Each group has a heading (in bold with a target) followed by a vertical gap):

```

5382 \renewcommand*{\glsgroupheading}[1]{%
5383     \par\noindent
5384     \textbf{\glsnavigation\hypertarget{##1}{\glsgrouptitle{##1}}}\par
5385     \indexspace}%
5386 }

```

\glstreeindent Length governing left indent for each level of the tree style.

```

5387 \newlength\glstreeindent
5388 \setlength{\glstreeindent}{10pt}

```

treenoname The **treenoname** glossary style is like the **tree** style, but doesn't print the name or symbol for sub-levels.

```
5389 \newglossarystyle{treenoname}{%
```

Set the paragraph indentation and skip:

```
5390 \renewenvironment{theglossary}%
5391 {\setlength{\parindent}{0pt}%
5392 \setlength{\parskip}{0pt plus 0.3pt}}%
5393 {}%
```

No header:

```
5394 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
5395 \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.

```
5396 \renewcommand{\glossaryentryfield}[5]{%
5397 \hangindent0pt\relax
5398 \parindent0pt\relax
5399 \glstentryitem{##1}\textbf{\glstarget{##1}{##2}}%
5400 \ifx\relax##4\relax
5401 \else
5402 \space{##4}%
5403 \fi
5404 \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.

```
5405 \renewcommand{\glossarysubentryfield}[6]{%
5406 \hangindent##1\glstreeindent\relax
5407 \parindent##1\glstreeindent\relax
5408 \ifnum##1=1\relax
5409 \glssubentryitem{##2}%
5410 \fi
5411 \glstarget{##2}{\strut}%
5412 ##4\glspostdescription\space ##6\par}%
```

Vertical gap between groups is the same as that used by indices:

```
5413 \renewcommand*{\glsgroupskip}{\indexspace}%
5414 }
```

treenonamegroup Like the **treenoname** style but the glossary groups have headings.

```
5415 \newglossarystyle{treenonamegroup}{%
```

Base it on the `glostytreetreenoname` style:

```
5416 \glossarystyle{treenoname}%
```

Give each group a heading:

```
5417 \renewcommand{\glsgroupheading}[1]{\par
5418 \noindent\textbf{\glsgetgrouptitle{##1}}\par\indexspace}%
5419 }
```

`treenonamehypergroup` The `treenonamehypergroup` style is like the `treenonamegroup` style, but has a set of links to the groups at the start of the glossary.

```
5420 \newglossarystyle{treenonamehypergroup}{%
```

Base it on the `glostyletreenoname` style:

```
5421 \glossarystyle{treenoname}%
```

Put navigation links to the groups at the start of the `theglossary` environment:

```
5422 \renewcommand*{\glossaryheader}{%
```

```
5423 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
5424 \renewcommand*{\glsgroupheading}[1]{%
```

```
5425 \par\noindent
```

```
5426 \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
```

```
5427 \indexspace}%
```

```
5428 }
```

`\glissetwidest` `\glissetwidest[<level>]{<text>}` sets the widest text for the given level. It is used by the `alttree` glossary styles to determine the indentation of each level.

```
5429 \newcommand*{\glissetwidest}[2][0]{%
```

```
5430 \expandafter\def\csname @glswidestname\romannumeral#1\endcsname{%
```

```
5431 #2}%
```

```
5432 }
```

`\@glswidestname` Initialise `\@glswidestname`.

```
5433 \newcommand*{\@glswidestname}{}
```

`alttree` The `alttree` glossary style is similar in style to the `tree` style, but the indentation is obtained from the width of `\@glswidestname` which is set using `\glissetwidest`.

```
5434 \newglossarystyle{alttree}{%
```

Redefine the `theglossary` environment.

```
5435 \renewenvironment{theglossary}%
```

```
5436 {\def\@gls@prevlevel{-1}%
```

```
5437 \mbox{}\par}%
```

```
5438 {\par}%
```

Set the header and group headers to nothing.

```
5439 \renewcommand*{\glossaryheader}{}%
```

```
5440 \renewcommand*{\glsgroupheading}[1]{}%
```

Redefine the way that the level 0 entries are displayed.

```
5441 \renewcommand{\glossaryentryfield}[5]{%
```

If the level hasn't changed, keep the same settings, otherwise change `\glstreeindent` accordingly.

```
5442 \ifnum\@gls@prevlevel=0\relax
```

```
5443 \else
```

Find out how big the indentation should be by measuring the widest entry.

```

5444      \settowidth{\glstreeindent}{\textbf{\@glswidestname\space}}%

```

Set the hangindent and paragraph indent.

```

5445      \hangindent\glstreeindent
5446      \parindent\glstreeindent
5447      \fi

```

Put the name to the left of the paragraph block.

```

5448      \makebox[0pt][r]{\makebox[\glstreeindent][l]{%
5449        \glstryitem{##1}\textbf{\glstarget{##1}{##2}}}%

```

If the symbol is missing, ignore it, otherwise put it in brackets.

```

5450      \ifx\relax##4\relax
5451      \else
5452        (##4)\space
5453      \fi

```

Do the description followed by the description terminator and location list.

```

5454      ##3\glspostdescription \space ##5\par

```

Set the previous level to 0.

```

5455      \def\@gls@prevlevel{0}%
5456    }%

```

Redefine the way sub-entries are displayed.

```

5457    \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.

```

5458      \ifnum##1=1\relax
5459        \glssubentryitem{##2}%
5460      \fi

```

If the level hasn't changed, keep the same settings, otherwise adjust \glstreeindent accordingly.

```

5461      \ifnum\@gls@prevlevel=##1\relax
5462      \else

```

Compute the widest entry for this level, or for level 0 if not defined for this level.

Store in \gls@tmplen

```

5463      \@ifundefined{\@glswidestname\romannumeral##1}{%
5464        \settowidth{\gls@tmplen}{\textbf{\@glswidestname\space}}{%
5465        \settowidth{\gls@tmplen}{\textbf{%
5466          \csname \@glswidestname\romannumeral##1\endcsname\space}}}%

```

Determine if going up or down a level

```

5467      \ifnum\@gls@prevlevel<##1\relax

```

Depth has increased, so add the width of the widest entry to \glstreeindent.

```

5468        \setlength\glstreeindent\gls@tmplen
5469        \addtolength\glstreeindent\parindent
5470        \parindent\glstreeindent
5471      \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`.

```

5472      \ifundefined{@glswidestname\romannumeral\@gls@prevlevel}{%
5473      \settowidth{\glstreeindent}{\textbf{%
5474      \@glswidestname\space}}}{%
5475      \settowidth{\glstreeindent}{\textbf{%
5476      \csname @glswidestname\romannumeral\@gls@prevlevel
5477      \endcsname\space}}}{%

```

Subtract this length from the previous level's paragraph indent and set to `\glstreeindent`.

```

5478      \addtolength\parindent{-\glstreeindent}%
5479      \setlength\glstreeindent\parindent
5480      \fi
5481      \fi

```

Set the hanging indentation.

```

5482      \hangindent\glstreeindent

```

Put the name to the left of the paragraph block

```

5483      \makebox[0pt][r]{\makebox[\@gls@tmplen][l]{%
5484      \textbf{\glstarget{##2}{##3}}}}%

```

If the symbol is missing, ignore it, otherwise put it in brackets.

```

5485      \ifx##5\relax\relax
5486      \else
5487      (##5)\space
5488      \fi

```

Do the description followed by the description terminator and location list.

```

5489      ##4\glspostdescription\space ##6\par

```

Set the previous level macro to the current level.

```

5490      \def\@gls@prevlevel{##1}%
5491      }%

```

Vertical gap between groups is the same as that used by indices:

```

5492      \renewcommand*{\glsgroupskip}{\indexspace}%
5493      }

```

almtreegroup Like the `almtree` style but the glossary groups have headings.

```

5494 \newglossarystyle{almtreegroup}{%

```

Base it on the `glostylealmtree` style:

```

5495 \glossarystyle{almtree}%

```

Give each group a heading.

```

5496 \renewcommand{\glsgroupheading}[1]{\par
5497 \def\@gls@prevlevel{-1}%
5498 \hangindent0pt\relax
5499 \parindent0pt\relax

```

```

5500 \textbf{\glsgetgrouptitle{##1}}\par\indexspace}%
5501 }

```

`almtreehypergroup` The `almtreehypergroup` style is like the `almtreegroup` style, but has a set of links to the groups at the start of the glossary.

```

5502 \newglossarystyle{almtreehypergroup}{%

```

Base it on the `glostylealmtree` style:

```

5503 \glossarystyle{almtree}%

```

Put the navigation links in the header

```

5504 \renewcommand*{\glossaryheader}{%
5505 \par
5506 \def\@gls@prevlevel{-1}%
5507 \hangindent0pt\relax
5508 \parindent0pt\relax
5509 \textbf{\glsnavigation}\par\indexspace}%

```

Put a hypertarget at the start of each group

```

5510 \renewcommand*{\glsgroupheading}[1]{%
5511 \par
5512 \def\@gls@prevlevel{-1}%
5513 \hangindent0pt\relax
5514 \parindent0pt\relax
5515 \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
5516 \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.

```

5517 \NeedsTeXFormat{LaTeX2e}
5518 \ProvidesPackage{glossaries-compatible-207}[2011/04/02 v1.0 (NLCT)]

```

`\GlsAddXdyAttribute` Adds an attribute in old format.

```

5519 \ifglsxindy
5520 \renewcommand*\GlsAddXdyAttribute[1]{%
5521 \edef\@xdyattributes{\@xdyattributes ^^J \string"#1\string"}%
5522 \expandafter\toks@\expandafter{\@xdylocref}%
5523 \edef\@xdylocref{\the\toks@ ^^J%
5524 (markup-locref
5525 :open \string"\string~n\string\setentrycounter
5526 {\noexpand\glscounter}%
5527 \expandafter\string\csname#1\endcsname
5528 \expandafter@gobble\string\{\string" ^^J
5529 :close \string"\expandafter@gobble\string\}\string" ^^J
5530 :attr \string"#1\string")}}

```

Only has an effect before \writeist:

5531 \fi

\GlsAddXdyCounters

```
5532 \renewcommand*\GlsAddXdyCounters[1]{%
5533   \GlossariesWarning{\string\GlsAddXdyCounters\space not available
5534     in compatibility mode.}%
5535 }
```

Add predefined attributes

```
5536 \GlsAddXdyAttribute{glsnumberformat}
5537 \GlsAddXdyAttribute{textrm}
5538 \GlsAddXdyAttribute{textsf}
5539 \GlsAddXdyAttribute{texttt}
5540 \GlsAddXdyAttribute{textbf}
5541 \GlsAddXdyAttribute{textmd}
5542 \GlsAddXdyAttribute{textit}
5543 \GlsAddXdyAttribute{textup}
5544 \GlsAddXdyAttribute{textsl}
5545 \GlsAddXdyAttribute{textsc}
5546 \GlsAddXdyAttribute{emph}
5547 \GlsAddXdyAttribute{glsnumber}
5548 \GlsAddXdyAttribute{hyper}
5549 \GlsAddXdyAttribute{hyperrm}
5549 \GlsAddXdyAttribute{hypersf}
5550 \GlsAddXdyAttribute{hypertt}
5551 \GlsAddXdyAttribute{hyperbf}
5552 \GlsAddXdyAttribute{hypermd}
5553 \GlsAddXdyAttribute{hyperit}
5554 \GlsAddXdyAttribute{hyperup}
5555 \GlsAddXdyAttribute{hypersl}
5556 \GlsAddXdyAttribute{hypersc}
5557 \GlsAddXdyAttribute{hyperemph}
```

\GlsAddXdyLocation Restore v2.07 definition:

```
5558 \ifglxindy
5559   \renewcommand*\GlsAddXdyLocation[2]{%
5560     \edef\@xdyuserlocationdefs{%
5561       \@xdyuserlocationdefs ^^J%
5562       (define-location-class \string"#1\string"^^J\space\space
5563         \space(#2))
5564     }%
5565     \edef\@xdyuserlocationnames{%
5566       \@xdyuserlocationnames^^J\space\space\space
5567       \string"#1\string"%
5568     }
5569 \fi
```

\@do@wrglossary

```
5570 \renewcommand*\@do@wrglossary[1]{%
```


Determine whether to use xindy or makeindex syntax

```
5571 \ifglxsindy
```

Need to determine if the formatting information starts with a (or) indicating a range.

```
5572 \expandafter\@glo@check@mkidxrangechar\@glsnumberformat\@nil
5573 \def\@glo@range{}%
5574 \expandafter\if\@glo@prefix(\relax
5575 \def\@glo@range{:open-range}%
5576 \else
5577 \expandafter\if\@glo@prefix)\relax
5578 \def\@glo@range{:close-range}%
5579 \fi
5580 \fi
```

Get the location and escape any special characters

```
5581 \protected@edef\@glslocref{\theglsentrycounter}%
5582 \@gls@checkmkidxchars\@glslocref
```

Write to the glossary file using xindy syntax.

```
5583 \glossary[\csname glo@#1@type\endcsname]{%
5584 (indexentry :tkey (\csname glo@#1@index\endcsname)
5585 :locref \string"\@glslocref\string" %
5586 :attr \string"\@glo@suffix\string" \@glo@range
5587 )
5588 }%
5589 \else
```

Convert the format information into the format required for makeindex

```
5590 \@set@glo@numformat\@glo@numfmt\@gls@counter\@glsnumberformat
```

Write to the glossary file using makeindex syntax.

```
5591 \glossary[\csname glo@#1@type\endcsname]{%
5592 \string\glossaryentry{\csname glo@#1@index\endcsname
5593 \@gls@encapchar\@glo@numfmt}{\theglsentrycounter}}%
5594 \fi
5595 }
```

\@set@glo@numformat Only had 3 arguments in v2.07

```
5596 \def\@set@glo@numformat#1#2#3{%
5597 \expandafter\@glo@check@mkidxrangechar#3\@nil
5598 \protected@edef#1{%
5599 \@glo@prefix setentrycounter[] {#2}%
5600 \expandafter\string\csname\@glo@suffix\endcsname
5601 }%
5602 \@gls@checkmkidxchars#1%
5603 }
```

\writeist Redefine \writeist back to the way it was in v2.07, but change \istfile to \glswrite.

```

5604 \ifglxindy
5605   \def\writeist{%
5606     \openout\glswrite=\istfilename
5607     \write\glswrite{;; xindy style file created by the glossaries
5608       package in compatible-2.07 mode}%
5609     \write\glswrite{;; for document '\jobname' on
5610       \the\year-\the\month-\the\day}%
5611     \write\glswrite{^^J; required styles^^J}
5612     \@for\@xdystyle:=\@xdyrequiredstyles\do{%
5613       \ifx\@xdystyle\@empty
5614       \else
5615         \protected@write\glswrite{{(require
5616           \string"\@xdystyle.xdy\string")}}%
5617       \fi
5618     }%
5619     \write\glswrite{^^J%
5620       ; list of allowed attributes (number formats)^^J}%
5621     \write\glswrite{(define-attributes ((\@xdyattributes)))}%
5622     \write\glswrite{^^J; user defined alphabets^^J}%
5623     \write\glswrite{\@xdyuseralphabets}%
5624     \write\glswrite{^^J; location class definitions^^J}%
5625     \protected@edef\@gls@roman{\@roman{0}\string"
5626       \string"roman-numbers-lowercase\string" :sep \string"}}%
5627     \@onelevel@sanitize\@gls@roman
5628     \edef\@tmp{\string" \string"roman-numbers-lowercase\string"
5629       :sep \string"}%
5630     \@onelevel@sanitize\@tmp
5631     \ifx\@tmp\@gls@roman
5632       \write\glswrite{(define-location-class
5633         \string"roman-page-numbers\string"^^J\space\space\space
5634         (\string"roman-numbers-lowercase\string")
5635         :min-range-length \@glsminrange)}}%
5636     \else
5637       \write\glswrite{(define-location-class
5638         \string"roman-page-numbers\string"^^J\space\space\space
5639         (:sep "\@gls@roman")
5640         :min-range-length \@glsminrange)}}%
5641     \fi
5642     \write\glswrite{(define-location-class
5643       \string"Roman-page-numbers\string"^^J\space\space\space
5644       (\string"roman-numbers-uppercase\string")
5645       :min-range-length \@glsminrange)}}%
5646     \write\glswrite{(define-location-class
5647       \string"arabic-page-numbers\string"^^J\space\space\space
5648       (\string"arabic-numbers\string")
5649       :min-range-length \@glsminrange)}}%
5650     \write\glswrite{(define-location-class
5651       \string"alpha-page-numbers\string"^^J\space\space\space
5652       (\string"alpha\string")

```

```

5653         :min-range-length \@glsminrange)}}%
5654 \write\glswrite{(define-location-class
5655   \string"Alpha-page-numbers\string"^^J\space\space\space
5656   (\string"ALPHA\string")
5657   :min-range-length \@glsminrange)}}%
5658 \write\glswrite{(define-location-class
5659   \string"Appendix-page-numbers\string"^^J\space\space\space
5660   (\string"ALPHA\string"
5661   :sep \string"\@glsAlphacompositor\string"
5662   \string"arabic-numbers\string")
5663   :min-range-length \@glsminrange)}}%
5664 \write\glswrite{(define-location-class
5665   \string"arabic-section-numbers\string"^^J\space\space\space
5666   (\string"arabic-numbers\string"
5667   :sep \string"\glscompositor\string"
5668   \string"arabic-numbers\string")
5669   :min-range-length \@glsminrange)}}%
5670 \write\glswrite{^^J; user defined location classes}%
5671 \write\glswrite{@xdyuserlocationdefs}%
5672 \write\glswrite{^^J; define cross-reference class^^J}%
5673 \write\glswrite{(define-crossref-class \string"see\string"
5674   :unverified )}%
5675 \write\glswrite{(markup-crossref-list
5676   :class \string"see\string"^^J\space\space\space
5677   :open \string"\string\glsseeformat\string"
5678   :close \string"{}\string")}%
5679 \write\glswrite{^^J; define the order of the location classes}%
5680 \write\glswrite{(define-location-class-order
5681   (\@xdylocationclassorder))}%
5682 \write\glswrite{^^J; define the glossary markup^^J}%
5683 \write\glswrite{(markup-index^^J\space\space\space
5684   :open \string"\string
5685   \glossarysection[\string\glossarytoctitle]{\string
5686   \glossarytitle}\string\glossarypreamble\string~n\string\begin
5687   {theglossary}\string\glossaryheader\string~n\string" ^^J\space
5688   \space\space:close \string"\expandafter\@gobble
5689   \string%\string~n\string
5690   \end{theglossary}\string\glossarypostamble
5691   \string~n\string" ^^J\space\space\space
5692   :tree)}}%
5693 \write\glswrite{(markup-letter-group-list
5694   :sep \string"\string\glsgroupskip\string~n\string")}%
5695 \write\glswrite{(markup-indexentry
5696   :open \string"\string\relax \string\glsresetentrylist
5697   \string~n\string")}%
5698 \write\glswrite{(markup-locclass-list :open
5699   \string"\glsopenbrace\string\glossaryentrynumbers
5700   \glsopenbrace\string\relax\space \string"^^J\space\space\space
5701   :sep \string", \string"

```

```

5702 :close \string"\glsclosebrace\glsclosebrace\string"))}%
5703 \write\glswrite{(markup-locoref-list
5704 :sep \string"\string\delimN\space\string"))}%
5705 \write\glswrite{(markup-range
5706 :sep \string"\string\delimR\space\string"))}%
5707 \@onelevel@sanitize\gls@suffixF
5708 \@onelevel@sanitize\gls@suffixFF
5709 \ifx\gls@suffixF\@empty
5710 \else
5711 \write\glswrite{(markup-range
5712 :close "\gls@suffixF" :length 1 :ignore-end))}%
5713 \fi
5714 \ifx\gls@suffixFF\@empty
5715 \else
5716 \write\glswrite{(markup-range
5717 :close "\gls@suffixFF" :length 2 :ignore-end))}%
5718 \fi
5719 \write\glswrite{^^J; define format to use for locations^^J}%
5720 \write\glswrite{\@xdylocoref}%
5721 \write\glswrite{^^J; define letter group list format^^J}%
5722 \write\glswrite{(markup-letter-group-list
5723 :sep \string"\string\glsgroupskip\string~n\string"))}%
5724 \write\glswrite{^^J; letter group headings^^J}%
5725 \write\glswrite{(markup-letter-group
5726 :open-head \string"\string\glsgroupheading
5727 \glsopenbrace\string"^^J\space\space\space
5728 :close-head \string"\glsclosebrace\string"))}%
5729 \write\glswrite{^^J; additional letter groups^^J}%
5730 \write\glswrite{\@xdylettergroups}%
5731 \write\glswrite{^^J; additional sort rules^^J}%
5732 \write\glswrite{\@dysortrules}%
5733 \noist}
5734 \else
5735 \edef\@gls@actualchar{\string?}
5736 \edef\@gls@encapchar{\string|}
5737 \edef\@gls@levelchar{\string!}
5738 \edef\@gls@quotechar{\string"}
5739 \def\writeist{\relax
5740 \openout\glswrite=\istfilename
5741 \write\glswrite{\expandafter\@gobble\string\% makeindex style file
5742 created by the glossaries package}
5743 \write\glswrite{\expandafter\@gobble\string\% for document
5744 '\jobname' on \the\year-\the\month-\the\day}
5745 \write\glswrite{actual '\@gls@actualchar'}
5746 \write\glswrite{encap '\@gls@encapchar'}
5747 \write\glswrite{level '\@gls@levelchar'}
5748 \write\glswrite{quote '\@gls@quotechar'}
5749 \write\glswrite{keyword \string"\string\glossaryentry\string"}
5750 \write\glswrite{preamble \string"\string\glossarysection[\string

```

```

5751     \glossarytoctitle]{\string\glossarytitle}\string
5752     \glossarypreamble\string\n\string\begin{theglossary}\string
5753     \glossaryheader\string\n\string"}
5754 \write\glswrite{postamble \string"\string%\string\n\string
5755     \end{theglossary}\string\glossarypostamble\string\n
5756     \string"}
5757 \write\glswrite{group_skip \string"\string\glsgroupskip\string\n
5758     \string"}
5759 \write\glswrite{item_0 \string"\string%\string\n\string"}
5760 \write\glswrite{item_1 \string"\string%\string\n\string"}
5761 \write\glswrite{item_2 \string"\string%\string\n\string"}
5762 \write\glswrite{item_01 \string"\string%\string\n\string"}
5763 \write\glswrite{item_x1
5764     \string"\string\relax \string\glsglresetentrylist\string\n
5765     \string"}
5766 \write\glswrite{item_12 \string"\string%\string\n\string"}
5767 \write\glswrite{item_x2
5768     \string"\string\relax \string\glsglresetentrylist\string\n
5769     \string"}
5770 \write\glswrite{delim_0 \string"\string\{\string
5771     \glossaryentrynumbers\string\{\string\relax \string"}
5772 \write\glswrite{delim_1 \string"\string\{\string
5773     \glossaryentrynumbers\string\{\string\relax \string"}
5774 \write\glswrite{delim_2 \string"\string\{\string
5775     \glossaryentrynumbers\string\{\string\relax \string"}
5776 \write\glswrite{delim_t \string"\string\}\string\}\string"}
5777 \write\glswrite{delim_n \string"\string\delimN \string"}
5778 \write\glswrite{delim_r \string"\string\delimR \string"}
5779 \write\glswrite{headings_flag 1}
5780 \write\glswrite{heading_prefix
5781     \string"\string\glsgroupheading\string\{\string"}
5782 \write\glswrite{heading_suffix
5783     \string"\string\}\string\relax
5784     \string\glsglresetentrylist \string"}
5785 \write\glswrite{symhead_positive \string"glssymbols\string"}
5786 \write\glswrite{numhead_positive \string"glsglnumbers\string"}
5787 \write\glswrite{page_compositor \string"glsglcompositor\string"}
5788 \@glsgl@escbsdq\glsgl@suffixF
5789 \@glsgl@escbsdq\glsgl@suffixFF
5790 \ifx\glsgl@suffixF\@empty
5791 \else
5792     \write\glswrite{suffix_2p \string"\glsgl@suffixF\string"}
5793 \fi
5794 \ifx\glsgl@suffixFF\@empty
5795 \else
5796     \write\glswrite{suffix_3p \string"\glsgl@suffixFF\string"}
5797 \fi
5798 \noist
5799 }

```

```
5800 \fi
```

```
\noist
```

```
5801 \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.

```
5802 \NeedsTeXFormat{LaTeX2e}
```

Package version number now in line with main glossaries package number but will only be updated when `glossaries-accsupp.sty` is modified.

```
5803 \ProvidesPackage{glossaries-accsupp}[2011/04/02 v3.0 (NLCT)]
```

```
5804 Experimental glossaries accessibility
```

Pass all options to glossaries:

```
5805 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{glossaries}}
```

Process options:

```
5806 \ProcessOptions
```

Required packages:

```
5807 \RequirePackage{glossaries}
```

```
5808 \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:

```
5809 \define@key{glossentry}{access}{%
```

```
5810 \def\@glo@access{#1}%
```

```
5811 }
```

textaccess The replacement text corresponding to the text key:

```
5812 \define@key{glossentry}{textaccess}{%
```

```
5813 \def\@glo@textaccess{#1}%
```

```
5814 }
```

firstaccess The replacement text corresponding to the first key:

```
5815 \define@key{glossentry}{firstaccess}{%
```

```
5816 \def\@glo@firstaccess{#1}%
```

```
5817 }
```

pluralaccess The replacement text corresponding to the plural key:

```
5818 \define@key{glossentry}{pluralaccess}{%
5819   \def\@glo@pluralaccess{#1}%
5820 }
```

firstpluralaccess The replacement text corresponding to the firstplural key:

```
5821 \define@key{glossentry}{firstpluralaccess}{%
5822   \def\@glo@firstpluralaccess{#1}%
5823 }
```

symbolaccess The replacement text corresponding to the symbol key:

```
5824 \define@key{glossentry}{symbolaccess}{%
5825   \def\@glo@symbolaccess{#1}%
5826 }
```

symbolpluralaccess The replacement text corresponding to the symbolplural key:

```
5827 \define@key{glossentry}{symbolpluralaccess}{%
5828   \def\@glo@symbolpluralaccess{#1}%
5829 }
```

descriptionaccess The replacement text corresponding to the description key:

```
5830 \define@key{glossentry}{descriptionaccess}{%
5831   \def\@glo@descaccess{#1}%
5832 }
```

descriptionpluralaccess The replacement text corresponding to the descriptionplural key:

```
5833 \define@key{glossentry}{descriptionpluralaccess}{%
5834   \def\@glo@descpluralaccess{#1}%
5835 }
```

shortaccess The replacement text corresponding to the short key:

```
5836 \define@key{glossentry}{shortaccess}{%
5837   \def\@glo@shortaccess{#1}%
5838 }
```

shortpluralaccess The replacement text corresponding to the shortplural key:

```
5839 \define@key{glossentry}{shortpluralaccess}{%
5840   \def\@glo@shortpluralaccess{#1}%
5841 }
```

longaccess The replacement text corresponding to the long key:

```
5842 \define@key{glossentry}{longaccess}{%
5843   \def\@glo@longaccess{#1}%
5844 }
```

longpluralaccess The replacement text corresponding to the longplural key:

```
5845 \define@key{glossentry}{longpluralaccess}{%
5846   \def\@glo@longpluralaccess{#1}%
5847 }
```

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.

```
5848 \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):

```
5849 \let\@gls@oldnewglossaryentryprehook\@newglossaryentryprehook
5850 \renewcommand*{\@newglossaryentryprehook}{%
5851   \@gls@oldnewglossaryentryprehook
5852   \def\@glo@access{\@glo@symbol}%
```

Initialise the other keys:

```
5853   \def\@glo@textaccess{\@glo@access}%
5854   \def\@glo@firstaccess{\@glo@access}%
5855   \def\@glo@pluralaccess{\@glo@textaccess}%
5856   \def\@glo@firstpluralaccess{\@glo@pluralaccess}%
5857   \def\@glo@symbolaccess{\relax}%
5858   \def\@glo@symbolpluralaccess{\@glo@symbolaccess}%
5859   \def\@glo@descaccess{\relax}%
5860   \def\@glo@descpluralaccess{\@glo@descaccess}%
5861   \def\@glo@shortaccess{\relax}%
5862   \def\@glo@shortpluralaccess{\@glo@shortaccess}%
5863   \def\@glo@longaccess{\relax}%
5864   \def\@glo@longpluralaccess{\@glo@longaccess}%
5865 }
```

Add to the end hook:

```
5866 \let\@gls@oldnewglossaryentryposthook\@newglossaryentryposthook
5867 \renewcommand*{\@newglossaryentryposthook}{%
5868   \@gls@oldnewglossaryentryposthook
```

Store the access information:

```
5869   \expandafter
5870     \protected@xdef\csname glo@\@glo@label @access\endcsname{%
5871       \@glo@access}%
5872   \expandafter
5873     \protected@xdef\csname glo@\@glo@label @textaccess\endcsname{%
5874       \@glo@textaccess}%
5875   \expandafter
5876     \protected@xdef\csname glo@\@glo@label @firstaccess\endcsname{%
5877       \@glo@firstaccess}%
5878   \expandafter
5879     \protected@xdef\csname glo@\@glo@label @pluralaccess\endcsname{%
5880       \@glo@pluralaccess}%
5881   \expandafter
5882     \protected@xdef\csname glo@\@glo@label @firstpluralaccess\endcsname{%
5883       \@glo@firstpluralaccess}%
5884   \expandafter
```



```

5885 \protected@xdef\csname glo@\@glo@label @symbolaccess\endcsname{%
5886 \@glo@symbolaccess}%
5887 \expandafter
5888 \protected@xdef\csname glo@\@glo@label @symbolpluralaccess\endcsname{%
5889 \@glo@symbolpluralaccess}%
5890 \expandafter
5891 \protected@xdef\csname glo@\@glo@label @descaccess\endcsname{%
5892 \@glo@descaccess}%
5893 \expandafter
5894 \protected@xdef\csname glo@\@glo@label @descpluralaccess\endcsname{%
5895 \@glo@descpluralaccess}%
5896 \expandafter
5897 \protected@xdef\csname glo@\@glo@label @shortaccess\endcsname{%
5898 \@glo@shortaccess}%
5899 \expandafter
5900 \protected@xdef\csname glo@\@glo@label @shortpluralaccess\endcsname{%
5901 \@glo@shortpluralaccess}%
5902 \expandafter
5903 \protected@xdef\csname glo@\@glo@label @longaccess\endcsname{%
5904 \@glo@longaccess}%
5905 \expandafter
5906 \protected@xdef\csname glo@\@glo@label @longpluralaccess\endcsname{%
5907 \@glo@longpluralaccess}%
5908 }

```

5.2 Accessing Replacement Text

`\glsentryaccess` Get the value of the access key for the entry with the given label:

```

5909 \newcommand*{\glsentryaccess}[1]{%
5910 \csname glo@#1@access\endcsname
5911 }

```

`\glsentrytextaccess` Get the value of the textaccess key for the entry with the given label:

```

5912 \newcommand*{\glsentrytextaccess}[1]{%
5913 \csname glo@#1@textaccess\endcsname
5914 }

```

`\glsentryfirstaccess` Get the value of the firstaccess key for the entry with the given label:

```

5915 \newcommand*{\glsentryfirstaccess}[1]{%
5916 \csname glo@#1@firstaccess\endcsname
5917 }

```

`\glsentrypluralaccess` Get the value of the pluralaccess key for the entry with the given label:

```

5918 \newcommand*{\glsentrypluralaccess}[1]{%
5919 \csname glo@#1@pluralaccess\endcsname
5920 }

```

`\glsentryfirstpluralaccess` Get the value of the firstpluralaccess key for the entry with the given label:

```

5921 \newcommand*{\glsentryfirstpluralaccess}[1]{%
5922   \csname glo@#1@firstpluralaccess\endcsname
5923 }

```

`\glsentrysymbolaccess` Get the value of the `symbolaccess` key for the entry with the given label:

```

5924 \newcommand*{\glsentrysymbolaccess}[1]{%
5925   \csname glo@#1@symbolaccess\endcsname
5926 }

```

`\glsentrysymbolpluralaccess` Get the value of the `symbolpluralaccess` key for the entry with the given label:

```

5927 \newcommand*{\glsentrysymbolpluralaccess}[1]{%
5928   \csname glo@#1@symbolpluralaccess\endcsname
5929 }

```

`\glsentrydescaccess` Get the value of the `descriptionaccess` key for the entry with the given label:

```

5930 \newcommand*{\glsentrydescaccess}[1]{%
5931   \csname glo@#1@descaccess\endcsname
5932 }

```

`\glsentrydescpluralaccess` Get the value of the `descriptionpluralaccess` key for the entry with the given label:

```

5933 \newcommand*{\glsentrydescpluralaccess}[1]{%
5934   \csname glo@#1@descaccess\endcsname
5935 }

```

`\glsentryshortaccess` Get the value of the `shortaccess` key for the entry with the given label:

```

5936 \newcommand*{\glsentryshortaccess}[1]{%
5937   \csname glo@#1@shortaccess\endcsname
5938 }

```

`\glsentryshortpluralaccess` Get the value of the `shortpluralaccess` key for the entry with the given label:

```

5939 \newcommand*{\glsentryshortpluralaccess}[1]{%
5940   \csname glo@#1@shortpluralaccess\endcsname
5941 }

```

`\glsentrylongaccess` Get the value of the `longaccess` key for the entry with the given label:

```

5942 \newcommand*{\glsentrylongaccess}[1]{%
5943   \csname glo@#1@longaccess\endcsname
5944 }

```

`\glsentrylongpluralaccess` Get the value of the `longpluralaccess` key for the entry with the given label:

```

5945 \newcommand*{\glsentrylongpluralaccess}[1]{%
5946   \csname glo@#1@longpluralaccess\endcsname
5947 }

```

`\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.)

```
5948 \newcommand*{\glsaccsupp}[2]{%
5949   \BeginAccSupp{ActualText=#1}#2\EndAccSupp{}%
5950 }
```

`\xglsaccsupp` Fully expands replacement text before calling `\glsaccsupp`

```
5951 \newcommand*{\xglsaccsupp}[2]{%
5952   \protected@edef\@gls@replacementtext{#1}%
5953   \expandafter\xglsaccsupp\expandafter{\@gls@replacementtext}{#2}%
5954 }
```

`\glsnameaccessdisplay` Displays the first argument with the accessibility text for the entry with the label given by the second argument (if set).

```
5955 \DeclareRobustCommand*{\glsnameaccessdisplay}[2]{%
5956   \protected@edef\@glo@access{\glsentryaccess{#2}}%
5957   \ifx\@glo@access\@gls@noaccess
5958     #1%
5959   \else
5960     \xglsaccsupp{\@glo@access}{#1}%
5961   \fi
5962 }
```

`\glsentrytextaccessdisplay` As above but for the `textaccess` replacement text.

```
5963 \DeclareRobustCommand*{\glsentrytextaccessdisplay}[2]{%
5964   \protected@edef\@glo@access{\glsentrytextaccess{#2}}%
5965   \ifx\@glo@access\@gls@noaccess
5966     #1%
5967   \else
5968     \xglsaccsupp{\@glo@access}{#1}%
5969   \fi
5970 }
```

`\glsentrypluralaccessdisplay` As above but for the `pluralaccess` replacement text.

```
5971 \DeclareRobustCommand*{\glsentrypluralaccessdisplay}[2]{%
5972   \protected@edef\@glo@access{\glsentrypluralaccess{#2}}%
5973   \ifx\@glo@access\@gls@noaccess
5974     #1%
5975   \else
5976     \xglsaccsupp{\@glo@access}{#1}%
5977   \fi
5978 }
```

`\glsentryfirstaccessdisplay` As above but for the `firstaccess` replacement text.

```
5979 \DeclareRobustCommand*{\glsentryfirstaccessdisplay}[2]{%
5980   \protected@edef\@glo@access{\glsentryfirstaccess{#2}}%
5981   \ifx\@glo@access\@gls@noaccess
5982     #1%
```

```

5983 \else
5984 \xglsaccsupp{\@glo@access}{#1}%
5985 \fi
5986 }

```

pluralaccessdisplay As above but for the firstpluralaccess replacement text.

```

5987 \DeclareRobustCommand*\glsfirstpluralaccessdisplay}[2]{%
5988 \protected@edef\@glo@access{\glsentryfirstpluralaccess{#2}}%
5989 \ifx\@glo@access\@gls@noaccess
5990 #1%
5991 \else
5992 \xglsaccsupp{\@glo@access}{#1}%
5993 \fi
5994 }

```

symbolaccessdisplay As above but for the symbolaccess replacement text.

```

5995 \DeclareRobustCommand*\glsymbolaccessdisplay}[2]{%
5996 \protected@edef\@glo@access{\glsentrysymbolaccess{#2}}%
5997 \ifx\@glo@access\@gls@noaccess
5998 #1%
5999 \else
6000 \xglsaccsupp{\@glo@access}{#1}%
6001 \fi
6002 }

```

pluralaccessdisplay As above but for the symbolpluralaccess replacement text.

```

6003 \DeclareRobustCommand*\glsymbolpluralaccessdisplay}[2]{%
6004 \protected@edef\@glo@access{\glsentrysymbolpluralaccess{#2}}%
6005 \ifx\@glo@access\@gls@noaccess
6006 #1%
6007 \else
6008 \xglsaccsupp{\@glo@access}{#1}%
6009 \fi
6010 }

```

descriptionaccessdisplay As above but for the descriptionaccess replacement text.

```

6011 \DeclareRobustCommand*\glsdescriptionaccessdisplay}[2]{%
6012 \protected@edef\@glo@access{\glsentrydescaccess{#2}}%
6013 \ifx\@glo@access\@gls@noaccess
6014 #1%
6015 \else
6016 \xglsaccsupp{\@glo@access}{#1}%
6017 \fi
6018 }

```

pluralaccessdisplay As above but for the descriptionpluralaccess replacement text.

```

6019 \DeclareRobustCommand*\glsdescriptionpluralaccessdisplay}[2]{%
6020 \protected@edef\@glo@access{\glsentrydescpluralaccess{#2}}%
6021 \ifx\@glo@access\@gls@noaccess

```

```

6022    #1%
6023    \else
6024        \xglsaccsupp{\@glo@access}{#1}%
6025    \fi
6026 }

```

`sshortaccessdisplay` As above but for the `shortaccess` replacement text.

```

6027 \DeclareRobustCommand*\glsshortaccessdisplay[2]{%
6028     \protected@edef\@glo@access{\glstryshortaccess{#2}}%
6029     \ifx\@glo@access\@gls@noaccess
6030         #1%
6031     \else
6032         \xglsaccsupp{\@glo@access}{#1}%
6033     \fi
6034 }

```

`pluralaccessdisplay` As above but for the `shortpluralaccess` replacement text.

```

6035 \DeclareRobustCommand*\glsshortpluralaccessdisplay[2]{%
6036     \protected@edef\@glo@access{\glstryshortpluralaccess{#2}}%
6037     \ifx\@glo@access\@gls@noaccess
6038         #1%
6039     \else
6040         \xglsaccsupp{\@glo@access}{#1}%
6041     \fi
6042 }

```

`lslongaccessdisplay` As above but for the `longaccess` replacement text.

```

6043 \DeclareRobustCommand*\glslongaccessdisplay[2]{%
6044     \protected@edef\@glo@access{\glstrylongaccess{#2}}%
6045     \ifx\@glo@access\@gls@noaccess
6046         #1%
6047     \else
6048         \xglsaccsupp{\@glo@access}{#1}%
6049     \fi
6050 }

```

`pluralaccessdisplay` As above but for the `longpluralaccess` replacement text.

```

6051 \DeclareRobustCommand*\glslongpluralaccessdisplay[2]{%
6052     \protected@edef\@glo@access{\glstrylongpluralaccess{#2}}%
6053     \ifx\@glo@access\@gls@noaccess
6054         #1%
6055     \else
6056         \xglsaccsupp{\@glo@access}{#1}%
6057     \fi
6058 }

```

`\glsaccessdisplay` Gets the replacement text corresponding to the named key given by the first argument and calls the appropriate command defined above.

```

6059 \DeclareRobustCommand*\glsaccessdisplay}[3]{%
6060   \@ifundefined{gls#1accessdisplay}%
6061   {%
6062     \PackageError{glossaries-accsupp}{No accessibility support
6063       for key ‘#1’}{%
6064     }%
6065   {%
6066     \csname gls#1accessdisplay\endcsname{#2}{#3}%
6067   }%
6068 }

```

\@gls@ Redefine \@gls@ to change the way the link text is defined

```

6069 \def\@gls@#1#2[#3]{%
6070   \glsdoifexists{#2}%
6071   {%
6072     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

6073   \def\@gls@link@opts{#1}%
6074   \def\@gls@link@label{#2}%

```

Determine what the link text should be (this is stored in \@glo@text). This is no longer expanded.

```

6075   \ifglsused{#2}%
6076   {%
6077     \def\@glo@text{\csname gls@\@glo@type @display\endcsname
6078       {\glsfirstaccessdisplay{\glsentryfirst{#2}}{#2}}%
6079       {\glsdescriptionaccessdisplay{\glsentrydesc{#2}}{#2}}%
6080       {\glsymbolaccessdisplay{\glsentrysymbol{#2}}{#2}}%
6081       {#3}}%
6082   }%
6083   {%
6084     \def\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
6085       {\glsfirstaccessdisplay{\glsentryfirst{#2}}{#2}}%
6086       {\glsdescriptionaccessdisplay{\glsentrydesc{#2}}{#2}}%
6087       {\glsymbolaccessdisplay{\glsentrysymbol{#2}}{#2}}%
6088       {#3}}%
6089   }%

```

Call \@gls@link. If footnote package option has been used, suppress hyperlink for first use.

```

6090   \ifglsused{#2}%
6091   {%
6092     \@gls@link[#1]{#2}{\@glo@text}%
6093   }%
6094   {%
6095     \gls@checkisacronymlist\@glo@type
6096     \ifthenelse{\boolean{glsisacronymlist}\AND
6097       \boolean{glsacrfootnote}}{\OR\NOT\boolean{glshyperfirst}}%
6098   }%

```

```

6099      \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
6100    }%
6101    {%
6102      \@gls@link[#1]{#2}{\@glo@text}%
6103    }%
6104  }%

```

Indicate that this entry has now been used

```

6105    \glsunset{#2}%
6106  }%
6107}

```

\@Gls@

```

6108 \def\@Gls@#1#2[#3]{%
6109   \glsdoifexists{#2}%
6110   {%
6111     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

6112   \def\@gls@link@opts{#1}%
6113   \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

```

6114   \ifglsused{#2}%
6115   {%
6116     \def\@glo@text{\csname gls@\@glo@type @display\endcsname
6117       {\glsfirstaccessdisplay{\Glsentrytext{#2}}{#2}}}%
6118     {\glsdescriptionaccessdisplay{\glsentrydesc{#2}}{#2}}%
6119     {\glsymbolaccessdisplay{\glsentrysymbol{#2}}{#2}}%
6120     {#3}}%
6121   }%
6122   {%
6123     \def\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
6124       {\glsfirstaccessdisplay{\Glsentryfirst{#2}}{#2}}%
6125       {\glsdescriptionaccessdisplay{\glsentrydesc{#2}}{#2}}%
6126       {\glsymbolaccessdisplay{\glsentrysymbol{#2}}{#2}}%
6127       {#3}}%
6128   }%

```

Call \@gls@link. If footnote package option has been used, suppress hyperlink for first use.

```

6129   \ifglsused{#2}%
6130   {%
6131     \@gls@link[#1]{#2}{\@glo@text}%
6132   }%
6133   {%
6134     \gls@checkisacronymlist\@glo@type
6135     \ifthenelse{\boolean{glsisacronymlist}}{\AND

```

```

6136      \boolean{glsacrfootnote}\) \OR\NOT\boolean{glshyperfirst}}}%
6137      {%
6138      \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
6139      }%
6140      {%
6141      \@gls@link[#1]{#2}{\@glo@text}%
6142      }%
6143      }%

```

Indicate that this entry has now been used

```

6144      \glsunset{#2}%
6145      }%
6146  }

```

\@GLS@

```

6147 \def\@GLS@#1#2[#3]{%
6148   \glsdoifexists{#2}{%
6149     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

6150   \def\@gls@link@opts{#1}%
6151   \def\@gls@link@label{#2}%

```

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

```

6152   \ifglsused{#2}%
6153   {%
6154     \def\@glo@text{\csname gls@\@glo@type @display\endcsname
6155       {\glsfirstaccessdisplay{\glsentryfirst{#2}}{#2}}}%
6156     {\glsdescriptionaccessdisplay{\glsentrydesc{#2}}{#2}}}%
6157     {\glsymbolaccessdisplay{\glsentrysymbol{#2}}{#2}}}%
6158     {#3}}}%
6159   }%
6160   {%
6161     \edef\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
6162       {\glsfirstaccessdisplay{\glsentryfirst{#2}}{#2}}}%
6163       {\glsdescriptionaccessdisplay{\glsentrydesc{#2}}{#2}}}%
6164       {\glsymbolaccessdisplay{\glsentrysymbol{#2}}{#2}}}%
6165       {#3}}}%
6166   }%

```

Call \@gls@link If footnote package option has been used, suppress hyperlink for first use.

```

6167   \ifglsused{#2}%
6168   {%
6169     \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}}%
6170   }%
6171   {%
6172     \gls@checkisacronymlist\@glo@type
6173     \ifthenelse{(\boolean{glsisacronymlist})\AND
6174       \boolean{glsacrfootnote}\) \OR\NOT\boolean{glshyperfirst}}{%
6175       \@gls@link[#1,hyper=false]{#2}{\MakeUppercase{\@glo@text}}}%

```



```

6176     }%
6177     {%
6178     \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
6179     }%
6180     }%

```

Indicate that this entry has now been used

```

6181     \glsunset{#2}%
6182     }%
6183 }

```

\@gls@pl@

```

6184 \def\@glspl@#1#2[#3]{%
6185   \glsdoifexists{#2}%
6186   {%
6187     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

6188     \def\@gls@link@opts{#1}%
6189     \def\@gls@link@label{#2}%

```

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

```

6190     \ifglsused{#2}%
6191     {%
6192       \def\@glo@text{\csname gls@\@glo@type @display\endcsname
6193         {\glspluralaccessdisplay{\glsentryplural{#2}}{#2}}%
6194         {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
6195         {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
6196         {#3}}%
6197     }%
6198     {%
6199       \def\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
6200         {\glsfirstpluralaccessdisplay{\glsentryfirstplural{#2}}{#2}}%
6201         {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
6202         {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
6203         {#3}}%
6204     }%

```

Call \@gls@link If footnote package option has been used, suppress hyperlink for first use.

```

6205     \ifglsused{#2}%
6206     {%
6207       \@gls@link[#1]{#2}{\@glo@text}%
6208     }%
6209     {%
6210       \gls@checkisacronymlist\@glo@type
6211       \ifthenelse{(\boolean{@glsisacronymlist}\AND
6212         \boolean{glsacrfootnote})\OR\NOT\boolean{glshyperfirst}}%
6213       {%
6214         \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
6215       }%

```

```

6216      {%
6217      \@gls@link[#1]{#2}{\@glo@text}%
6218      }%
6219      }%

```

Indicate that this entry has now been used

```

6220      \glsunset{#2}%
6221      }%
6222 }

```

\@Glspl@

```

6223 \def\@Glspl@#1#2[#3]{%
6224   \glsdoifexists{#2}%
6225   {%
6226     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

6227     \def\@gls@link@opts{#1}%
6228     \def\@gls@link@label{#2}%

```

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

```

6229     \ifglsused{#2}%
6230     {%
6231       \def\@glo@text{\csname gls@\@glo@type @display\endcsname
6232         {\glspluralaccessdisplay{\Glsentryplural{#2}}{#2}}%
6233         {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
6234         {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
6235         {#3}}%
6236     }%
6237     {%
6238       \def\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
6239         {\glsfirstpluralaccessdisplay{\Glsentryfirstplural{#2}}{#2}}%
6240         {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
6241         {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
6242         {#3}}%
6243     }%

```

Call \@gls@link If footnote package option has been used, suppress hyperlink for first use.

```

6244     \ifglsused{#2}%
6245     {%
6246       \@gls@link[#1]{#2}{\@glo@text}%
6247     }%
6248     {%
6249       \ifthenelse{\equal{\@glo@type}{\acronymtype}\and
6250         \boolean{glsacrfootnote}}%
6251       {%
6252         \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
6253       }%
6254       {%
6255         \@gls@link[#1]{#2}{\@glo@text}%

```

```

6256     }%
6257 }%

```

Indicate that this entry has now been used

```

6258     \glsunset{#2}%
6259 }%
6260 }

```

\@GLSp1@

```

6261 \def\@GLSp1@#1#2[#3]{%
6262   \glsdoifexists{#2}%
6263   {%
6264     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

6265     \def\@gls@link@opts{#1}%
6266     \def\@gls@link@label{#2}%

```

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

```

6267     \ifglsused{#2}%
6268     {%
6269       \def\@glo@text{\csname gls@\@glo@type @display\endcsname
6270         {\glspluralaccessdisplay{\glsentryplural{#2}}{#2}}%
6271         {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
6272         {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
6273         {#3}}%
6274     }%
6275     {%
6276       \def\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
6277         {\glsfirstpluralaccessdisplay{\glsentryfirstplural{#2}}{#2}}%
6278         {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
6279         {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
6280         {#3}}%
6281     }%

```

Call \@gls@link If footnote package option has been used, suppress hyperlink for first use.

```

6282     \ifglsused{#2}%
6283     {%
6284       \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
6285     }%
6286     {%
6287       \gls@checkisacronymlist\@glo@type
6288       \ifthenelse{(\boolean{glsisacronymlist}\AND
6289         \boolean{glsacrfootnote})\OR\not\boolean{glshyperfirst}}%
6290       {%
6291         \@gls@link[#1,hyper=false]{#2}{\MakeUppercase{\@glo@text}}%
6292       }%
6293     }%
6294     \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
6295     }%

```

```

6296     }%
        Indicate that this entry has now been used
6297     \glsunset{#2}%
6298     }%
6299 }

```

\@acrshort

```

6300 \def\@acrshort#1#2[#3]{%
6301   \glsdoifexists{#2}%
6302   {%
6303     \edef\@glo@type{\glsentrytype{#2}}%
        Determine what the link text should be (this is stored in \@glo@text)
6304     \def\@glo@text{%
6305       \glsshortaccessdisplay{\glsentryshort{#2}}{#2}%
6306     }%
        Call \@gls@link
6307     \@gls@link[#1]{#2}{\acronymfont{\@glo@text}#3}%
6308   }%
6309 }

```

\@Acrshort

```

6310 \def\@Acrshort#1#2[#3]{%
6311   \glsdoifexists{#2}%
6312   {%
6313     \edef\@glo@type{\glsentrytype{#2}}%
        Determine what the link text should be (this is stored in \@glo@text)
6314     \def\@glo@text{%
6315       \glsshortaccessdisplay{\Glsentryshort{#2}}{#2}%
6316     }%
        Call \@gls@link
6317     \@gls@link[#1]{#2}{\acronymfont{\@glo@text}#3}%
6318   }%
6319 }

```

\@ACRshort

```

6320 \def\@ACRshort#1#2[#3]{%
6321   \glsdoifexists{#2}%
6322   {%
6323     \edef\@glo@type{\glsentrytype{#2}}%
        Determine what the link text should be (this is stored in \@glo@text)
6324     \def\@glo@text{%
6325       \glsshortaccessdisplay{\MakeUppercase{\glsentryshort{#2}}}{#2}%
6326     }%

```

```

        Call \@gls@link
6327     \@gls@link[#1]{#2}{\acronymfont{\@glo@text#3}}%
6328   }%
6329 }

```

\@acrlong

```

6330 \def\@acrlong#1#2[#3]{%
6331   \glsdoifexists{#2}%
6332   {%
6333     \edef\@glo@type{\glsentrytype{#2}}%
        Determine what the link text should be (this is stored in \@glo@text)
6334     \def\@glo@text{%
6335       \glslongaccessdisplay{\glsentrylong{#2}}{#2}%
6336     }%
        Call \@gls@link
6337     \@gls@link[#1]{#2}{\@glo@text#3}%
6338   }%
6339 }

```

\@Acrlong

```

6340 \def\@Acrlong#1#2[#3]{%
6341   \glsdoifexists{#2}%
6342   {%
6343     \edef\@glo@type{\glsentrytype{#2}}%
        Determine what the link text should be (this is stored in \@glo@text)
6344     \def\@glo@text{%
6345       \glslongaccessdisplay{\Glsentrylong{#2}}{#2}%
6346     }%
        Call \@gls@link
6347     \@gls@link[#1]{#2}{\@glo@text#3}%
6348   }%
6349 }

```

\@ACRlong

```

6350 \def\@ACRlong#1#2[#3]{%
6351   \glsdoifexists{#2}%
6352   {%
6353     \edef\@glo@type{\glsentrytype{#2}}%
        Determine what the link text should be (this is stored in \@glo@text)
6354     \def\@glo@text{%
6355       \glslongaccessdisplay{\MakeUppercase{\glsentrylong{#2}}}{#2}%
6356     }%
        Call \@gls@link
6357     \@gls@link[#1]{#2}{\@glo@text#3}%
6358   }%
6359 }

```

5.3 Displaying the Glossary

Entries within the glossary or list of acronyms are now formatted via `\accsuppglossaryentryfield` and `\accsuppglossarysubentryfield`.

@glossaryentryfield

```
6360 \ifglxindy
6361   \renewcommand*{\@glossaryentryfield}{%
6362     \string\accsuppglossaryentryfield}
6363 \else
6364   \renewcommand*{\@glossaryentryfield}{%
6365     \string\accsuppglossaryentryfield}
6366 \fi
```

ossarysubentryfield

```
6367 \ifglxindy
6368   \renewcommand*{\@glossarysubentryfield}{%
6369     \string\accsuppglossarysubentryfield}
6370 \else
6371   \renewcommand*{\@glossarysubentryfield}{%
6372     \string\accsuppglossarysubentryfield}
6373 \fi
```

pglossaryentryfield

```
6374 \newcommand*{\accsuppglossaryentryfield}[5]{%
6375   \glossaryentryfield{#1}%
6376   {\glxnameaccessdisplay{#2}{#1}}%
6377   {\glxdescriptionaccessdisplay{#3}{#1}}%
6378   {\glxsymbolaccessdisplay{#4}{#1}}{#5}%
6379 }
```

ossarysubentryfield

```
6380 \newcommand*{\accsuppglossarysubentryfield}[6]{%
6381   \glossaryentryfield{#1}{#2}%
6382   {\glxnameaccessdisplay{#3}{#2}}%
6383   {\glxdescriptionaccessdisplay{#4}{#2}}%
6384   {\glxsymbolaccessdisplay{#5}{#2}}{#6}%
6385 }
```

5.4 Acronyms

Use `\newacronymhook` to modify the key list to set the access text to the long version by default.

```
6386 \renewcommand*{\newacronymhook}{%
6387   \edef\@gls@keylist{shortaccess=\the\glxlongtok,%
6388     \the\glxkeylisttok}%
6389   \expandafter\glxkeylisttok\expandafter{\@gls@keylist}%
6390 }
```

efaultNewAcronymDef Modify default style to use access text:

```

6391 \renewcommand*{\DefaultNewAcronymDef}{%
6392   \edef\@do@newglossaryentry{%
6393     \noexpand\newglossaryentry{\the\glslabeltok}%
6394     {%
6395       type=\acronymtype,%
6396       name={\the\glsshorttok},%
6397       description={\the\glslongtok},%
6398       descriptionaccess=\relax,
6399       text={\the\glsshorttok},%
6400       access={\noexpand\@glo@textaccess},%
6401       sort={\the\glsshorttok},%
6402       short={\the\glsshorttok},%
6403       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6404       shortaccess={\the\glslongtok},%
6405       long={\the\glslongtok},%
6406       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6407       descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6408       first={\noexpand\glslongaccessdisplay
6409         {\the\glslongtok}{\the\glslabeltok}\space
6410         (\noexpand\glsshortaccessdisplay
6411           {\the\glsshorttok}{\the\glslabeltok})},%
6412       plural={\the\glsshorttok\acrpluralsuffix},%
6413       firstplural={\noexpand\glslongpluralaccessdisplay
6414         {\noexpand\@glo@longpl}{\the\glslabeltok}\space
6415         (\noexpand\glsshortpluralaccessdisplay
6416           {\noexpand\@glo@shortpl}{\the\glslabeltok})},%
6417       firstaccess=\relax,
6418       firstpluralaccess=\relax,
6419       textaccess={\noexpand\@glo@shortaccess},%
6420       \the\glskeylisttok
6421     }%
6422   }%
6423   \@do@newglossaryentry
6424 }

```

otnoteNewAcronymDef

```

6425 \renewcommand*{\DescriptionFootnoteNewAcronymDef}{%
6426   \edef\@do@newglossaryentry{%
6427     \noexpand\newglossaryentry{\the\glslabeltok}%
6428     {%
6429       type=\acronymtype,%
6430       name={\noexpand\acronymfont{\the\glsshorttok}},%
6431       sort={\the\glsshorttok},%
6432       text={\the\glsshorttok},%
6433       short={\the\glsshorttok},%
6434       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6435       shortaccess={\the\glslongtok},%
6436       long={\the\glslongtok},%

```

```

6437     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6438     access={\noexpand\@glo@textaccess},%
6439     plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6440     symbol={\the\glslongtok},%
6441     symbolplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6442     firstpluralaccess=\relax,
6443     textaccess={\noexpand\@glo@shortaccess},%
6444     \the\glskeylisttok
6445 }%
6446 }%
6447 \@do@newglossaryentry
6448 }

```

ptionNewAcronymDef

```

6449 \renewcommand*{\DescriptionNewAcronymDef}{%
6450   \edef\@do@newglossaryentry{%
6451     \noexpand\newglossaryentry{\the\glslabeltok}%
6452     {%
6453       type=\acronymtype,%
6454       name={\noexpand
6455         \acrnameformat{\the\glsshorttok}{\the\glslongtok}},%
6456       access={\noexpand\@glo@textaccess},%
6457       sort={\the\glsshorttok},%
6458       short={\the\glsshorttok},%
6459       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6460       shortaccess={\the\glslongtok},%
6461       long={\the\glslongtok},%
6462       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6463       first={\the\glslongtok},%
6464       firstaccess=\relax,
6465       firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6466       text={\the\glsshorttok},%
6467       textaccess={\the\glslongtok},%
6468       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6469       symbol={\noexpand\@glo@text},%
6470       symbolaccess={\noexpand\@glo@textaccess},%
6471       symbolplural={\noexpand\@glo@plural},%
6472       firstpluralaccess=\relax,
6473       textaccess={\noexpand\@glo@shortaccess},%
6474       \the\glskeylisttok}%
6475   }%
6476   \@do@newglossaryentry
6477 }

```

otnoteNewAcronymDef

```

6478 \renewcommand*{\FootnoteNewAcronymDef}{%
6479   \edef\@do@newglossaryentry{%
6480     \noexpand\newglossaryentry{\the\glslabeltok}%
6481     {%

```



```

6482     type=\acronymtype,%
6483     name={\noexpand\acronymfont{\the\glsshorttok}},%
6484     sort={\the\glsshorttok},%
6485     text={\the\glsshorttok},%
6486     textaccess={\the\glslongtok},%
6487     access={\noexpand\@glo@textaccess},%
6488     plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6489     short={\the\glsshorttok},%
6490     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6491     long={\the\glslongtok},%
6492     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6493     description={\the\glslongtok},%
6494     descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6495     \the\glskeylisttok
6496   }%
6497 }%
6498 \@do@newglossaryentry
6499 }

```

\SmallNewAcronymDef

```

6500 \renewcommand*{\SmallNewAcronymDef}{%
6501   \edef\@do@newglossaryentry{%
6502     \noexpand\newglossaryentry{\the\glslabeltok}%
6503     {%
6504       type=\acronymtype,%
6505       name={\noexpand\acronymfont{\the\glsshorttok}},%
6506       access={\noexpand\@glo@symbolaccess},%
6507       sort={\the\glsshorttok},%
6508       short={\the\glsshorttok},%
6509       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6510       shortaccess={\the\glslongtok},%
6511       long={\the\glslongtok},%
6512       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6513       text={\noexpand\@glo@short},%
6514       textaccess={\noexpand\@glo@shortaccess},%
6515       plural={\noexpand\@glo@shortpl},%
6516       first={\the\glslongtok},%
6517       firstaccess=\relax,%
6518       firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6519       description={\noexpand\@glo@first},%
6520       descriptionplural={\noexpand\@glo@firstplural},%
6521       symbol={\the\glsshorttok},%
6522       symbolaccess={\the\glslongtok},%
6523       symbolplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6524       \the\glskeylisttok
6525     }%
6526   }%
6527   \@do@newglossaryentry
6528 }

```

The following are kept for compatibility with versions before 3.0:

```
\glsshortaccesskey
6529 \newcommand*{\glsshortaccesskey}{\glsshortkey access}%

\shortpluralaccesskey
6530 \newcommand*{\glsshortpluralaccesskey}{\glsshortpluralkey access}%

\glslongaccesskey
6531 \newcommand*{\glslongaccesskey}{\glslongkey access}%

\longpluralaccesskey
6532 \newcommand*{\glslongpluralaccesskey}{\glslongpluralkey access}%
```

5.5 Debugging Commands

```
\showglonameaccess
6533 \newcommand*{\showglonameaccess}[1]{%
6534 \expandafter\show\csname glo@#1@textaccess\endcsname
6535 }

\showglotextaccess
6536 \newcommand*{\showglotextaccess}[1]{%
6537 \expandafter\show\csname glo@#1@textaccess\endcsname
6538 }

\showglopluralaccess
6539 \newcommand*{\showglopluralaccess}[1]{%
6540 \expandafter\show\csname glo@#1@pluralaccess\endcsname
6541 }

\showglofirstaccess
6542 \newcommand*{\showglofirstaccess}[1]{%
6543 \expandafter\show\csname glo@#1@firstaccess\endcsname
6544 }

\glofirstpluralaccess
6545 \newcommand*{\showglofirstpluralaccess}[1]{%
6546 \expandafter\show\csname glo@#1@firstpluralaccess\endcsname
6547 }

\showglosymbolaccess
6548 \newcommand*{\showglosymbolaccess}[1]{%
6549 \expandafter\show\csname glo@#1@symbolaccess\endcsname
6550 }
```

osymbolpluralaccess

```
6551 \newcommand*{\showglosymbolpluralaccess}[1]{%
6552   \expandafter\show\csname glo@#1@symbolpluralaccess\endcsname
6553 }
```

\showglodescaccess

```
6554 \newcommand*{\showglodescaccess}[1]{%
6555   \expandafter\show\csname glo@#1@descaccess\endcsname
6556 }
```

glodescpluralaccess

```
6557 \newcommand*{\showglodescpluralaccess}[1]{%
6558   \expandafter\show\csname glo@#1@descpluralaccess\endcsname
6559 }
```

\showgloshortaccess

```
6560 \newcommand*{\showgloshortaccess}[1]{%
6561   \expandafter\show\csname glo@#1@shortaccess\endcsname
6562 }
```

loshortpluralaccess

```
6563 \newcommand*{\showgloshortpluralaccess}[1]{%
6564   \expandafter\show\csname glo@#1@shortpluralaccess\endcsname
6565 }
```

\showglolongaccess

```
6566 \newcommand*{\showglolongaccess}[1]{%
6567   \expandafter\show\csname glo@#1@longaccess\endcsname
6568 }
```

glolongpluralaccess

```
6569 \newcommand*{\showglolongpluralaccess}[1]{%
6570   \expandafter\show\csname glo@#1@longpluralaccess\endcsname
6571 }
```

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.

```
6572 \NeedsTeXFormat{LaTeX2e}
6573 \ProvidesPackage{glossaries-babel}[2009/04/16 v1.2 (NLCT)]
```

English:

```
6574 \@ifundefined{captionsenglish}{%}{%
6575   \addto\captionsenglish{%
6576     \renewcommand*{\glossaryname}{Glossary}%
6577     \renewcommand*{\acronymname}{Acronyms}%
6578     \renewcommand*{\entryname}{Notation}%
6579     \renewcommand*{\descriptionname}{Description}%
6580     \renewcommand*{\symbolname}{Symbol}%
6581     \renewcommand*{\pagelistname}{Page List}%
6582     \renewcommand*{\glssymbolsgroupname}{Symbols}%
6583     \renewcommand*{\glsnumbersgroupname}{Numbers}%
6584 }%
6585 }
6586 \@ifundefined{captionsamerican}{%}{%
6587   \addto\captionsamerican{%
6588     \renewcommand*{\glossaryname}{Glossary}%
6589     \renewcommand*{\acronymname}{Acronyms}%
6590     \renewcommand*{\entryname}{Notation}%
6591     \renewcommand*{\descriptionname}{Description}%
6592     \renewcommand*{\symbolname}{Symbol}%
6593     \renewcommand*{\pagelistname}{Page List}%
6594     \renewcommand*{\glssymbolsgroupname}{Symbols}%
6595     \renewcommand*{\glsnumbersgroupname}{Numbers}%
6596 }%
6597 }
6598 \@ifundefined{captionsaustralian}{%}{%
6599   \addto\captionsaustralian{%
6600     \renewcommand*{\glossaryname}{Glossary}%
6601     \renewcommand*{\acronymname}{Acronyms}%
6602     \renewcommand*{\entryname}{Notation}%
6603     \renewcommand*{\descriptionname}{Description}%
6604     \renewcommand*{\symbolname}{Symbol}%
6605     \renewcommand*{\pagelistname}{Page List}%
6606     \renewcommand*{\glssymbolsgroupname}{Symbols}%
6607     \renewcommand*{\glsnumbersgroupname}{Numbers}%
6608 }%
6609 }
6610 \@ifundefined{captionsbritish}{%}{%
6611   \addto\captionsbritish{%
6612     \renewcommand*{\glossaryname}{Glossary}%
6613     \renewcommand*{\acronymname}{Acronyms}%
6614     \renewcommand*{\entryname}{Notation}%
6615     \renewcommand*{\descriptionname}{Description}%
6616     \renewcommand*{\symbolname}{Symbol}%
6617     \renewcommand*{\pagelistname}{Page List}%
6618     \renewcommand*{\glssymbolsgroupname}{Symbols}%
6619     \renewcommand*{\glsnumbersgroupname}{Numbers}%
6620 }%
6621 \@ifundefined{captionscanadian}{%}{%
```

```

6622 \addto\captionscanadian{%
6623   \renewcommand*{\glossaryname}{Glossary}%
6624   \renewcommand*{\acronymname}{Acronyms}%
6625   \renewcommand*{\entryname}{Notation}%
6626   \renewcommand*{\descriptionname}{Description}%
6627   \renewcommand*{\symbolname}{Symbol}%
6628   \renewcommand*{\pagelistname}{Page List}%
6629   \renewcommand*{\glssymbolsgroupname}{Symbols}%
6630   \renewcommand*{\glsnumbersgroupname}{Numbers}%
6631 }%
6632 }
6633 \@ifundefined{captionsnewzealand}{}{%
6634   \addto\captionscanadian{%
6635     \renewcommand*{\glossaryname}{Glossary}%
6636     \renewcommand*{\acronymname}{Acronyms}%
6637     \renewcommand*{\entryname}{Notation}%
6638     \renewcommand*{\descriptionname}{Description}%
6639     \renewcommand*{\symbolname}{Symbol}%
6640     \renewcommand*{\pagelistname}{Page List}%
6641     \renewcommand*{\glssymbolsgroupname}{Symbols}%
6642     \renewcommand*{\glsnumbersgroupname}{Numbers}%
6643 }%
6644 }
6645 \@ifundefined{captionsUKenglish}{}{%
6646   \addto\captionscanadian{%
6647     \renewcommand*{\glossaryname}{Glossary}%
6648     \renewcommand*{\acronymname}{Acronyms}%
6649     \renewcommand*{\entryname}{Notation}%
6650     \renewcommand*{\descriptionname}{Description}%
6651     \renewcommand*{\symbolname}{Symbol}%
6652     \renewcommand*{\pagelistname}{Page List}%
6653     \renewcommand*{\glssymbolsgroupname}{Symbols}%
6654     \renewcommand*{\glsnumbersgroupname}{Numbers}%
6655 }%
6656 }
6657 \@ifundefined{captionsUSenglish}{}{%
6658   \addto\captionscanadian{%
6659     \renewcommand*{\glossaryname}{Glossary}%
6660     \renewcommand*{\acronymname}{Acronyms}%
6661     \renewcommand*{\entryname}{Notation}%
6662     \renewcommand*{\descriptionname}{Description}%
6663     \renewcommand*{\symbolname}{Symbol}%
6664     \renewcommand*{\pagelistname}{Page List}%
6665     \renewcommand*{\glssymbolsgroupname}{Symbols}%
6666     \renewcommand*{\glsnumbersgroupname}{Numbers}%
6667 }%
6668 }

```

German (quite a few variations were suggested for German; I settled on the following):

```

6669 \@ifundefined{captionsgerman}{\}%
6670   \addto\captionsgerman{%
6671     \renewcommand*{\glossaryname}{Glossar}%
6672     \renewcommand*{\acronymname}{Akronyme}%
6673     \renewcommand*{\entryname}{Bezeichnung}%
6674     \renewcommand*{\descriptionname}{Beschreibung}%
6675     \renewcommand*{\symbolname}{Symbol}%
6676     \renewcommand*{\pagelistname}{Seiten}%
6677     \renewcommand*{\glssymbolsgroupname}{Symbole}%
6678     \renewcommand*{\glsnumbersgroupname}{Zahlen}}
6679 }

```

ngerman is identical to German:

```

6680 \@ifundefined{captionsgerman}{\}%
6681   \addto\captionsgerman{%
6682     \renewcommand*{\glossaryname}{Glossar}%
6683     \renewcommand*{\acronymname}{Akronyme}%
6684     \renewcommand*{\entryname}{Bezeichnung}%
6685     \renewcommand*{\descriptionname}{Beschreibung}%
6686     \renewcommand*{\symbolname}{Symbol}%
6687     \renewcommand*{\pagelistname}{Seiten}%
6688     \renewcommand*{\glssymbolsgroupname}{Symbole}%
6689     \renewcommand*{\glsnumbersgroupname}{Zahlen}}
6690 }

```

Italian:

```

6691 \@ifundefined{captionssitalian}{\}%
6692   \addto\captionssitalian{%
6693     \renewcommand*{\glossaryname}{Glossario}%
6694     \renewcommand*{\acronymname}{Acronimi}%
6695     \renewcommand*{\entryname}{Nomenclatura}%
6696     \renewcommand*{\descriptionname}{Descrizione}%
6697     \renewcommand*{\symbolname}{Simbolo}%
6698     \renewcommand*{\pagelistname}{Elenco delle pagine}%
6699     \renewcommand*{\glssymbolsgroupname}{Simboli}%
6700     \renewcommand*{\glsnumbersgroupname}{Numeri}}
6701 }

```

Dutch:

```

6702 \@ifundefined{captionsdutch}{\}%
6703   \addto\captionsdutch{%
6704     \renewcommand*{\glossaryname}{Woordenlijst}%
6705     \renewcommand*{\acronymname}{Acroniemen}%
6706     \renewcommand*{\entryname}{Benaming}%
6707     \renewcommand*{\descriptionname}{Beschrijving}%
6708     \renewcommand*{\symbolname}{Symbool}%
6709     \renewcommand*{\pagelistname}{Pagina's}%
6710     \renewcommand*{\glssymbolsgroupname}{Symbolen}%

```

```

6711 \renewcommand*{\glsnumbersgroupname}{Cijfers}}
6712 }

Spanish:

6713 \@ifundefined{captionsspanish}{}{%
6714 \addto\captionsspanish{%
6715 \renewcommand*{\glossaryname}{Glosario}%
6716 \renewcommand*{\acronymname}{Siglas}%
6717 \renewcommand*{\entryname}{Entrada}%
6718 \renewcommand*{\descriptionname}{Descripci\`on}%
6719 \renewcommand*{\symbolname}{S\`{i}mbolo}%
6720 \renewcommand*{\pagelistname}{Lista de p\`aginas}%
6721 \renewcommand*{\glssymbolsgroupname}{S\`{i}mbolos}%
6722 \renewcommand*{\glsnumbersgroupname}{N\`umeros}}
6723 }

French:

6724 \@ifundefined{captionsfrench}{}{%
6725 \addto\captionsfrench{%
6726 \renewcommand*{\glossaryname}{Glossaire}%
6727 \renewcommand*{\acronymname}{Acronymes}%
6728 \renewcommand*{\entryname}{Terme}%
6729 \renewcommand*{\descriptionname}{Description}%
6730 \renewcommand*{\symbolname}{Symbole}%
6731 \renewcommand*{\pagelistname}{Pages}%
6732 \renewcommand*{\glssymbolsgroupname}{Symboles}%
6733 \renewcommand*{\glsnumbersgroupname}{Nombres}}
6734 }

6735 \@ifundefined{captionsfrenchb}{}{%
6736 \addto\captionsfrenchb{%
6737 \renewcommand*{\glossaryname}{Glossaire}%
6738 \renewcommand*{\acronymname}{Acronymes}%
6739 \renewcommand*{\entryname}{Terme}%
6740 \renewcommand*{\descriptionname}{Description}%
6741 \renewcommand*{\symbolname}{Symbole}%
6742 \renewcommand*{\pagelistname}{Pages}%
6743 \renewcommand*{\glssymbolsgroupname}{Symboles}%
6744 \renewcommand*{\glsnumbersgroupname}{Nombres}}
6745 }

6746 \@ifundefined{captionsfrancais}{}{%
6747 \addto\captionsfrancais{%
6748 \renewcommand*{\glossaryname}{Glossaire}%
6749 \renewcommand*{\acronymname}{Acronymes}%
6750 \renewcommand*{\entryname}{Terme}%
6751 \renewcommand*{\descriptionname}{Description}%
6752 \renewcommand*{\symbolname}{Symbole}%
6753 \renewcommand*{\pagelistname}{Pages}%
6754 \renewcommand*{\glssymbolsgroupname}{Symboles}%
6755 \renewcommand*{\glsnumbersgroupname}{Nombres}}
6756 }

```

Danish:

```
6757 \@ifundefined{captionsdanish}{%  
6758   \addto\captionsdanish{%  
6759     \renewcommand*{\glossaryname}{Ordliste}%  
6760     \renewcommand*{\acronymname}{Akronymer}%  
6761     \renewcommand*{\entryname}{Symbolforklaring}%  
6762     \renewcommand*{\descriptionname}{Beskrivelse}%  
6763     \renewcommand*{\symbolname}{Symbol}%  
6764     \renewcommand*{\pagelistname}{Side}%  
6765     \renewcommand*{\glssymbolsgroupname}{Symboler}%  
6766     \renewcommand*{\glsnumbersgroupname}{Tal}}  
6767 }
```

Irish:

```
6768 \@ifundefined{captionsirish}{%  
6769   \addto\captionsirish{%  
6770     \renewcommand*{\glossaryname}{Gluais}%  
6771     \renewcommand*{\acronymname}{Acrainmneacha}%
```

wasn't sure whether to go for Nóta (Note), Ciall ('Meaning', 'sense') or Brí ('Meaning'). In the end I chose Ciall.

```
6772     \renewcommand*{\entryname}{Ciall}%  
6773     \renewcommand*{\descriptionname}{Tuairisc}%
```

Again, not sure whether to use Comhartha/Comharthaí or Siombail/Siombaile, so have chosen the former.

```
6774     \renewcommand*{\symbolname}{Comhartha}%  
6775     \renewcommand*{\glssymbolsgroupname}{Comhartha\'}  
6776     \renewcommand*{\pagelistname}{Leathanaigh}%  
6777     \renewcommand*{\glsnumbersgroupname}{Uimhreacha}}  
6778 }
```

Hungarian:

```
6779 \@ifundefined{captionsmagyar}{%  
6780   \addto\captionsmagyar{%  
6781     \renewcommand*{\glossaryname}{Sz\'ojegyz\'ek}%  
6782     \renewcommand*{\acronymname}{Bet\H uszavak}%  
6783     \renewcommand*{\entryname}{Kifejez\'es}%  
6784     \renewcommand*{\descriptionname}{Magyar\'azat}%  
6785     \renewcommand*{\symbolname}{Jel\'ol\'es}%  
6786     \renewcommand*{\pagelistname}{Oldalsz\'am}%  
6787     \renewcommand*{\glssymbolsgroupname}{Jelek}%  
6788     \renewcommand*{\glsnumbersgroupname}{Sz\'amjegyek}%  
6789   }  
6790 }  
6791 \@ifundefined{captionshungarian}{%  
6792   \addto\captionshungarian{%  
6793     \renewcommand*{\glossaryname}{Sz\'ojegyz\'ek}%  
6794     \renewcommand*{\acronymname}{Bet\H uszavak}%  
6795     \renewcommand*{\entryname}{Kifejez\'es}%  
6796     \renewcommand*{\descriptionname}{Magyar\'azat}%
```



```

6797 \renewcommand*{\symbolname}{Jel\`ol\`es}%
6798 \renewcommand*{\pagelistname}{Oldalsz\`am}%
6799 \renewcommand*{\glssymbolsgroupname}{Jelek}%
6800 \renewcommand*{\glsnumbersgroupname}{Sz\`amjegyek}%
6801 }
6802 }

Polish
6803 \@ifundefined{captionspolish}{}{%
6804 \addto\captionspolish{%
6805 \renewcommand*{\glossaryname}{S\`lownik termin\`ow}%
6806 \renewcommand*{\acronymname}{Skr\`ot}%
6807 \renewcommand*{\entryname}{Termin}%
6808 \renewcommand*{\descriptionname}{Opis}%
6809 \renewcommand*{\symbolname}{Symbol}%
6810 \renewcommand*{\pagelistname}{Strony}%
6811 \renewcommand*{\glssymbolsgroupname}{Symbole}%
6812 \renewcommand*{\glsnumbersgroupname}{Liczby}}
6813 }

Brazilian
6814 \@ifundefined{captionsbrazil}{}{%
6815 \addto\captionsbrazil{%
6816 \renewcommand*{\glossaryname}{Gloss\`ario}%
6817 \renewcommand*{\acronymname}{Siglas}%
6818 \renewcommand*{\entryname}{Nota\`c c\`ao}%
6819 \renewcommand*{\descriptionname}{Descri\`c c\`ao}%
6820 \renewcommand*{\symbolname}{S\`imbolo}%
6821 \renewcommand*{\pagelistname}{Lista de P\`aginas}%
6822 \renewcommand*{\glssymbolsgroupname}{S\`imbolos}%
6823 \renewcommand*{\glsnumbersgroupname}{N\`umeros}%
6824 }%
6825 }

```

6.2 Polyglossia Captions

```

6826 \NeedsTeXFormat{LaTeX2e}
6827 \ProvidesPackage{glossaries-polyglossia}[2009/11/09 v1.0 (NLCT)]

English:
6828 \@ifundefined{captionsenglish}{}{%
6829 \expandafter\toks@\expandafter{\captionsenglish
6830 \renewcommand*{\glossaryname}{\textenglish{Glossary}}%
6831 \renewcommand*{\acronymname}{\textenglish{Acronyms}}%
6832 \renewcommand*{\entryname}{\textenglish{Notation}}%
6833 \renewcommand*{\descriptionname}{\textenglish{Description}}%
6834 \renewcommand*{\symbolname}{\textenglish{Symbol}}%
6835 \renewcommand*{\pagelistname}{\textenglish{Page List}}%
6836 \renewcommand*{\glssymbolsgroupname}{\textenglish{Symbols}}%
6837 \renewcommand*{\glsnumbersgroupname}{\textenglish{Numbers}}%
6838 }%

```

```

6839 \edef\captionseenglish{\the\toks@}%
6840 }

```

German:

```

6841 \@ifundefined{captionsgerman}{}{%
6842 \expandafter\toks@\expandafter{\captionsgerman
6843 \renewcommand*{\glossaryname}{\textgerman{Glossar}}}%
6844 \renewcommand*{\acronymname}{\textgerman{Akronyme}}}%
6845 \renewcommand*{\entryname}{\textgerman{Bezeichnung}}}%
6846 \renewcommand*{\descriptionname}{\textgerman{Beschreibung}}}%
6847 \renewcommand*{\symbolname}{\textgerman{Symbol}}}%
6848 \renewcommand*{\pagelistname}{\textgerman{Seiten}}}%
6849 \renewcommand*{\glssymbolsgroupname}{\textgerman{Symbole}}}%
6850 \renewcommand*{\glsnumbersgroupname}{\textgerman{Zahlen}}}%
6851 }%
6852 \edef\captionsgerman{\the\toks@}%
6853 }

```

Italian:

```

6854 \@ifundefined{captionssitalian}{}{%
6855 \expandafter\toks@\expandafter{\captionssitalian
6856 \renewcommand*{\glossaryname}{\textitalian{Glossario}}}%
6857 \renewcommand*{\acronymname}{\textitalian{Acronimi}}}%
6858 \renewcommand*{\entryname}{\textitalian{Nomenclatura}}}%
6859 \renewcommand*{\descriptionname}{\textitalian{Descrizione}}}%
6860 \renewcommand*{\symbolname}{\textitalian{Simbolo}}}%
6861 \renewcommand*{\pagelistname}{\textitalian{Elenco delle pagine}}}%
6862 \renewcommand*{\glssymbolsgroupname}{\textitalian{Simboli}}}%
6863 \renewcommand*{\glsnumbersgroupname}{\textitalian{Numeri}}}%
6864 }%
6865 \edef\captionssitalian{\the\toks@}%
6866 }

```

Dutch:

```

6867 \@ifundefined{captionsdutch}{}{%
6868 \expandafter\toks@\expandafter{\captionsdutch
6869 \renewcommand*{\glossaryname}{\textdutch{Woordenlijst}}}%
6870 \renewcommand*{\acronymname}{\textdutch{Acroniemen}}}%
6871 \renewcommand*{\entryname}{\textdutch{Benaming}}}%
6872 \renewcommand*{\descriptionname}{\textdutch{Beschrijving}}}%
6873 \renewcommand*{\symbolname}{\textdutch{Symbool}}}%
6874 \renewcommand*{\pagelistname}{\textdutch{Pagina's}}}%
6875 \renewcommand*{\glssymbolsgroupname}{\textdutch{Symbolen}}}%
6876 \renewcommand*{\glsnumbersgroupname}{\textdutch{Cijfers}}}%
6877 }%
6878 \edef\captionsdutch{\the\toks@}%
6879 }

```

Spanish:

```

6880 \@ifundefined{captionssspanish}{}{%
6881 \expandafter\toks@\expandafter{\captionssspanish
6882 \renewcommand*{\glossaryname}{\textspanish{Glosario}}}%

```

```

6883 \renewcommand*{\acronymname}{\textspanish{Siglas}}%
6884 \renewcommand*{\entryname}{\textspanish{Entrada}}%
6885 \renewcommand*{\descriptionname}{\textspanish{Descripci' on}}%
6886 \renewcommand*{\symbolname}{\textspanish{S\'\i mbolo}}%
6887 \renewcommand*{\pagelistname}{\textspanish{Lista de p\ ' aginas}}%
6888 \renewcommand*{\glssymbolsgroupname}{\textspanish{S\'\i mbolos}}%
6889 \renewcommand*{\glslnumbersgroupname}{\textspanish{N\ ' umeros}}%
6890 }%
6891 \edef\captionsspanish{\the\toks@}%
6892 }

```

French:

```

6893 \@ifundefined{captionsfrench}{}{%
6894 \expandafter\toks@\expandafter{\captionsfrench
6895 \renewcommand*{\glossaryname}{\textfrench{Glossaire}}%
6896 \renewcommand*{\acronymname}{\textfrench{Acronymes}}%
6897 \renewcommand*{\entryname}{\textfrench{Terme}}%
6898 \renewcommand*{\descriptionname}{\textfrench{Description}}%
6899 \renewcommand*{\symbolname}{\textfrench{Symbole}}%
6900 \renewcommand*{\pagelistname}{\textfrench{Pages}}%
6901 \renewcommand*{\glssymbolsgroupname}{\textfrench{Symboles}}%
6902 \renewcommand*{\glslnumbersgroupname}{\textfrench{Nombres}}%
6903 }%
6904 \edef\captionsfrench{\the\toks@}%
6905 }

```

Danish:

```

6906 \@ifundefined{captionsdanish}{}{%
6907 \expandafter\toks@\expandafter{\captionsdanish
6908 \renewcommand*{\glossaryname}{\textdanish{Ordliste}}%
6909 \renewcommand*{\acronymname}{\textdanish{Akronymer}}%
6910 \renewcommand*{\entryname}{\textdanish{Symbolforklaring}}%
6911 \renewcommand*{\descriptionname}{\textdanish{Beskrivelse}}%
6912 \renewcommand*{\symbolname}{\textdanish{Symbol}}%
6913 \renewcommand*{\pagelistname}{\textdanish{Side}}%
6914 \renewcommand*{\glssymbolsgroupname}{\textdanish{Symboler}}%
6915 \renewcommand*{\glslnumbersgroupname}{\textdanish{Tal}}%
6916 }%
6917 \edef\captionsdanish{\the\toks@}%
6918 }

```

Irish:

```

6919 \@ifundefined{captionsirish}{}{%
6920 \expandafter\toks@\expandafter{\captionsirish
6921 \renewcommand*{\glossaryname}{\textirish{Gluais}}%
6922 \renewcommand*{\acronymname}{\textirish{Acrainmneacha}}%
6923 \renewcommand*{\entryname}{\textirish{Ciall}}%
6924 \renewcommand*{\descriptionname}{\textirish{Tuirisc}}%
6925 \renewcommand*{\symbolname}{\textirish{Comhartha}}%
6926 \renewcommand*{\glssymbolsgroupname}{\textirish{Comhartha\ ' \i}}%
6927 \renewcommand*{\pagelistname}{\textirish{Leathanaigh}}%

```

```

6928 \renewcommand*{\glnumbersgroupname}{\textirish{Uimhreacha}}}%
6929 }%
6930 \edef\captionisirish{\the\toks@}%
6931 }

```

Hungarian:

```

6932 \@ifundefined{captionsmagyar}{}{%
6933 \expandafter\toks@\expandafter{\captionsmagyar
6934 \renewcommand*{\glossaryname}{\textmagyar{Sz\'ojegyz\'ek}}}%
6935 \renewcommand*{\acronymname}{\textmagyar{Bet\H uszavak}}}%
6936 \renewcommand*{\entryname}{\textmagyar{Kifejez\'es}}}%
6937 \renewcommand*{\descriptionname}{\textmagyar{Magyar\'azat}}}%
6938 \renewcommand*{\symbolname}{\textmagyar{Jel\'ol\'es}}}%
6939 \renewcommand*{\pagelistname}{\textmagyar{Oldalsz\'am}}}%
6940 \renewcommand*{\glssymbolsgroupname}{\textmagyar{Jelek}}}%
6941 \renewcommand*{\glnumbersgroupname}{\textmagyar{Sz\'amjegyek}}}%
6942 }%
6943 \edef\captionsmagyar{\the\toks@}%
6944 }

```

Polish

```

6945 \@ifundefined{captionspolish}{}{%
6946 \expandafter\toks@\expandafter{\captionspolish
6947 \renewcommand*{\glossaryname}{\textpolish{S\l ownnik termin\'ow}}}%
6948 \renewcommand*{\acronymname}{\textpolish{Skr\'ot}}}%
6949 \renewcommand*{\entryname}{\textpolish{Termin}}}%
6950 \renewcommand*{\descriptionname}{\textpolish{Opis}}}%
6951 \renewcommand*{\symbolname}{\textpolish{Symbol}}}%
6952 \renewcommand*{\pagelistname}{\textpolish{Strony}}}%
6953 \renewcommand*{\glssymbolsgroupname}{\textpolish{Symbole}}}%
6954 \renewcommand*{\glnumbersgroupname}{\textpolish{Liczby}}}%
6955 }%
6956 \edef\captionspolish{\the\toks@}%
6957 }

```

Portugues

```

6958 \@ifundefined{captionsportuges}{}{%
6959 \expandafter\toks@\expandafter{\captionsportuges
6960 \renewcommand*{\glossaryname}{\textportuges{Gloss\'ario}}}%
6961 \renewcommand*{\acronymname}{\textportuges{Siglas}}}%
6962 \renewcommand*{\entryname}{\textportuges{Nota\c c\~ao}}}%
6963 \renewcommand*{\descriptionname}{\textportuges{Descri\c c\~ao}}}%
6964 \renewcommand*{\symbolname}{\textportuges{S\'imbolo}}}%
6965 \renewcommand*{\pagelistname}{\textportuges{Lista de P\'aginas}}}%
6966 \renewcommand*{\glssymbolsgroupname}{\textportuges{S\'imbolos}}}%
6967 \renewcommand*{\glnumbersgroupname}{\textportuges{N\'umeros}}}%
6968 }%
6969 \edef\captionsportuges{\the\toks@}%
6970 }

```

6.3 Brazilian Dictionary

This is a dictionary file provided by Thiago de Melo for use with the package.

```
6971 \ProvidesDictionary{glossaries-dictionary}{Brazilian}
```

Provide Brazilian translations:

```
6972 \providetranslation{Glossary}{Gloss\`ario}
6973 \providetranslation{Acronyms}{Siglas}
6974 \providetranslation{Notation (glossaries)}{Nota\c c\~ao}
6975 \providetranslation{Description (glossaries)}{Descri\c c\~ao}
6976 \providetranslation{Symbol (glossaries)}{S\`imbolo}
6977 \providetranslation{Page List (glossaries)}{Lista de P\`aginas}
6978 \providetranslation{Symbols (glossaries)}{S\`imbolos}
6979 \providetranslation{Numbers (glossaries)}{N\`umeros}
```

6.4 Danish Dictionary

This is a dictionary file provided for use with the package.

```
6980 \ProvidesDictionary{glossaries-dictionary}{Danish}
```

Provide Danish translations:

```
6981 \providetranslation{Glossary}{Ordliste}
6982 \providetranslation{Acronyms}{Akronymer}
6983 \providetranslation{Notation (glossaries)}{Symbolforklaring}
6984 \providetranslation{Description (glossaries)}{Beskrivelse}
6985 \providetranslation{Symbol (glossaries)}{Symbol}
6986 \providetranslation{Page List (glossaries)}{Side}
6987 \providetranslation{Symbols (glossaries)}{Symboler}
6988 \providetranslation{Numbers (glossaries)}{Tal}
```

6.5 Dutch Dictionary

This is a dictionary file provided for use with the package.

```
6989 \ProvidesDictionary{glossaries-dictionary}{Dutch}
```

Provide Dutch translations:

```
6990 \providetranslation{Glossary}{Woordenlijst}
6991 \providetranslation{Acronyms}{Acroniemen}
6992 \providetranslation{Notation (glossaries)}{Benaming}
6993 \providetranslation{Description (glossaries)}{Beschrijving}
6994 \providetranslation{Symbol (glossaries)}{Symbool}
6995 \providetranslation{Page List (glossaries)}{Pagina's}
6996 \providetranslation{Symbols (glossaries)}{Symbolen}
6997 \providetranslation{Numbers (glossaries)}{Cijfers}
```

6.6 English Dictionary

This is a dictionary file provided for use with the package.

```
6998 \ProvidesDictionary{glossaries-dictionary}{English}
```

Provide English translations:

```
6999 \providetranslation{Glossary}{Glossary}
7000 \providetranslation{Acronyms}{Acronyms}
7001 \providetranslation{Notation (glossaries)}{Notation}
7002 \providetranslation{Description (glossaries)}{Description}
7003 \providetranslation{Symbol (glossaries)}{Symbol}
7004 \providetranslation{Page List (glossaries)}{Page List}
7005 \providetranslation{Symbols (glossaries)}{Symbols}
7006 \providetranslation{Numbers (glossaries)}{Numbers}
```

6.7 French Dictionary

This is a dictionary file provided for use with the package.

```
7007 \ProvidesDictionary{glossaries-dictionary}{French}
```

Provide French translations:

```
7008 \providetranslation{Glossary}{Glossaire}
7009 \providetranslation{Acronyms}{Acronymes}
7010 \providetranslation{Notation (glossaries)}{Terme}
7011 \providetranslation{Description (glossaries)}{Description}
7012 \providetranslation{Symbol (glossaries)}{Symbole}
7013 \providetranslation{Page List (glossaries)}{Pages}
7014 \providetranslation{Symbols (glossaries)}{Symboles}
7015 \providetranslation{Numbers (glossaries)}{Nombres}
```

6.8 German Dictionary

This is a dictionary file provided for use with the package.

```
7016 \ProvidesDictionary{glossaries-dictionary}{German}
```

Provide German translations (quite a few variations were suggested for German; I settled on the following):

```
7017 \providetranslation{Glossary}{Glossar}
7018 \providetranslation{Acronyms}{Akronyme}
7019 \providetranslation{Notation (glossaries)}{Bezeichnung}
7020 \providetranslation{Description (glossaries)}{Beschreibung}
7021 \providetranslation{Symbol (glossaries)}{Symbol}
7022 \providetranslation{Page List (glossaries)}{Seiten}
7023 \providetranslation{Symbols (glossaries)}{Symbole}
7024 \providetranslation{Numbers (glossaries)}{Zahlen}
```

6.9 Irish Dictionary

This is a dictionary file provided for use with the package.

```
7025 \ProvidesDictionary{glossaries-dictionary}{Irish}
```

Provide Irish translations:

```
7026 \providetranslation{Glossary}{Gluais}
7027 \providetranslation{Acronyms}{Acrainmneacha}
```

```

7028 \providetranslation{Notation (glossaries)}{Ciall}
7029 \providetranslation{Description (glossaries)}{Tuairisc}
7030 \providetranslation{Symbol (glossaries)}{Comhartha}
7031 \providetranslation{Page List (glossaries)}{Leathanaigh}
7032 \providetranslation{Symbols (glossaries)}{Comhartha'\{i}}
7033 \providetranslation{Numbers (glossaries)}{Uimhreacha}

```

6.10 Italian Dictionary

This is a dictionary file provided for use with the package.

```
7034 \ProvidesDictionary{glossaries-dictionary}{Italian}
```

Provide Italian translations:

```

7035 \providetranslation{Glossary}{Glossario}
7036 \providetranslation{Acronyms}{Acronimi}
7037 \providetranslation{Notation (glossaries)}{Nomenclatura}
7038 \providetranslation{Description (glossaries)}{Descrizione}
7039 \providetranslation{Symbol (glossaries)}{Simbolo}
7040 \providetranslation{Page List (glossaries)}{Elenco delle pagine}
7041 \providetranslation{Symbols (glossaries)}{Simboli}
7042 \providetranslation{Numbers (glossaries)}{Numeri}

```

6.11 Magyar Dictionary

This is a dictionary file provided for use with the package.

```
7043 \ProvidesDictionary{glossaries-dictionary}{Magyar}
```

Provide translations:

```

7044 \providetranslation{Glossary}{Sz\'ojegyz\'ek}
7045 \providetranslation{Acronyms}{Bet\H uszavak}
7046 \providetranslation{Notation (glossaries)}{Kifejez\'es}
7047 \providetranslation{Description (glossaries)}{Magyar\'azat}
7048 \providetranslation{Symbol (glossaries)}{Jel\'ol\'es}
7049 \providetranslation{Page List (glossaries)}{Oldalsz\'am}
7050 \providetranslation{Symbols (glossaries)}{Jelek}
7051 \providetranslation{Numbers (glossaries)}{Sz\'amjegyek}

```

6.12 Polish Dictionary

This is a dictionary file provided for use with the package.

```
7052 \ProvidesDictionary{glossaries-dictionary}{Polish}
```

Provide Polish translations:

```

7053 \providetranslation{Glossary}{S\lownik termin\'ow}
7054 \providetranslation{Acronyms}{Skr\'ot}
7055 \providetranslation{Notation (glossaries)}{Termin}
7056 \providetranslation{Description (glossaries)}{Opis}
7057 \providetranslation{Symbol (glossaries)}{Symbol}
7058 \providetranslation{Page List (glossaries)}{Strony}

```

```

7059 \providetranslation{Symbols (glossaries)}{Symbole}
7060 \providetranslation{Numbers (glossaries)}{Liczby}

```

6.13 Serbian Dictionary

This dictionary was provided by Zoran Filipovic.

```

7061 \ProvidesDictionary{glossaries-dictionary}{Serbian}
7062 \providetranslation{Glossary}{Mali re\vnik}
7063 \providetranslation{Acronyms}{Skra\'cenice}
7064 \providetranslation{Notation (glossaries)}{Oznaka}
7065 \providetranslation{Description (glossaries)}{Opis}
7066 \providetranslation{Symbol (glossaries)}{Simbol}
7067 \providetranslation{Page List (glossaries)}{Stranica}
7068 \providetranslation{Symbols (glossaries)}{Simboli}
7069 \providetranslation{Numbers (glossaries)}{Brojevi}

```

6.14 Spanish Dictionary

This is a dictionary file provided for use with the package.

```

7070 \ProvidesDictionary{glossaries-dictionary}{Spanish}

```

Provide Spanish translations:

```

7071 \providetranslation{Glossary}{Glosario}
7072 \providetranslation{Acronyms}{Siglas}
7073 \providetranslation{Notation (glossaries)}{Entrada}
7074 \providetranslation{Description (glossaries)}{Descripci\'on}
7075 \providetranslation{Symbol (glossaries)}{S\'mbolo}
7076 \providetranslation{Page List (glossaries)}{Lista de p\'aginas}
7077 \providetranslation{Symbols (glossaries)}{S\'mbolos}
7078 \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>4</u>
\@glossaryseclabel	<u>4</u>
\@glossarysecstar	<u>4</u>
\@ACRlong	<u>229</u>
\@ACRshort	<u>228</u>
\@Acrlong	<u>229</u>
\@Acrshort	<u>228</u>
\@GLS@	<u>70</u> , <u>224</u>
\@GLSpl	<u>73</u>
\@GLSpl@	<u>227</u>
\@Gls@	<u>69</u> , <u>223</u>
\@Glspl@	<u>72</u> , <u>226</u>
\@acrlong	<u>229</u>
\@acrshort	<u>228</u>
\@addtoacronymlists	<u>10</u>
\@delimN	<u>138</u>
\@delimR	<u>138</u>
\@disable@onlypremakeg	<u>17</u>
\@disable@premakecs	<u>17</u>
\@disabled@gl saddxdycounters ..	<u>27</u>
\@do@seeglossary	<u>125</u>
\@do@wrglossary	<u>123</u> , <u>208</u>
\@glo@seeautonumberlist	<u>5</u>
\@glo@storeentry	<u>51</u>
\@glo@types	<u>38</u>
\@glossary	<u>122</u>
\@glossary@default@style	<u>5</u>
\@glossaryentryfield	<u>51</u> , <u>230</u>
\@glossarysection	<u>25</u>
\@glossarysubentryfield ..	<u>51</u> , <u>230</u>
\@glS	<u>68</u>
\@glS@	<u>68</u> , <u>222</u>
\@glS@@link	<u>57</u>
\@glS@addpredefinedattributes ..	<u>29</u>
\@glS@checkactual	<u>64</u>
\@glS@checkbar	<u>63</u>
\@glS@checkescactual	<u>62</u>
\@glS@checkescbar	<u>62</u>
\@glS@checkesclevel	<u>63</u>
\@glS@checkescquote	<u>61</u>
\@glS@checklevel	<u>64</u>
\@glS@checkmkidxchars	<u>60</u>
\@glS@checkquote	<u>61</u>
\@glS@codepage	<u>35</u>
\@glS@counterwithin	<u>7</u>
\@glS@escbsdq	<u>60</u>
\@glS@fixbraces	<u>125</u>
\@glS@getcounter	<u>40</u>
\@glS@getcounterprefix	<u>124</u>
\@glS@hypergroup	<u>167</u>
\@glS@ifinlist	<u>27</u>
\@glS@link	<u>57</u>
\@glS@loadlist	<u>6</u>
\@glS@loadlong	<u>6</u>
\@glS@loadsuper	<u>6</u>
\@glS@loadtree	<u>6</u>
\@glS@missingnumberlist	<u>45</u>
\@glS@noaccess	<u>216</u>
\@glS@onlypremakeg	<u>17</u>
\@glS@pl@	<u>225</u>
\@glS@renewglossary	<u>123</u>
\@glS@sanitizedesc	<u>12</u>
\@glS@sanitizename	<u>12</u>
\@glS@sanitizesymbol	<u>12</u>
\@glS@saveentrycounter	<u>58</u>
\@glS@setcounter	<u>40</u>
\@glS@setupsort@def	<u>8</u>
\@glS@setupsort@standard	<u>7</u>
\@glS@setupsort@use	<u>8</u>
\@glS@tmpb	<u>61</u>
\@glS@toc	<u>26</u>
\@glS@updatechecked	<u>61</u>
\@glS@xdy@Lclass@Alpha-page-numbers	<u>31</u>
\@glS@xdy@Lclass@Appendix-page-numbers	<u>31</u>
\@glS@xdy@Lclass@Roman-page-numbers	<u>30</u>
\@glS@xdy@Lclass@alpha-page-numbers	<u>30</u>
\@glS@xdy@Lclass@arabic-page-numbers	<u>30</u>
\@glS@xdy@Lclass@arabic-section-numbers	<u>31</u>

<code>\addglossarytocaptions</code>	19	<code>\delimN</code>	23 , 137
<code>\addto</code>	19	<code>\delimR</code>	23 , 137
<code>align (environment)</code>	58	<code>description (environment)</code>	170
<code>altlist (style)</code>	171	<code>description (key)</code>	41
<code>altlistgroup (style)</code>	171	<code>description (option)</code>	14
<code>altlisthypergroup (style)</code>	172	<code>descriptionaccess (key)</code>	215
<code>atlong4col (style)</code>	177	<code>\DescriptionDUANewAcronymDef</code>	148
<code>atlong4colborder (style)</code>	178	<code>\DescriptionFootnoteNewAcronymDef</code>	146 , 231
<code>atlong4colheader (style)</code>	178	<code>\descriptionname</code>	18
<code>atlong4colheaderborder (style)</code>	178	<code>\DescriptionNewAcronymDef</code>	149 , 232
<code>atlongragged4col (style)</code>	182	<code>descriptionplural (key)</code>	41
<code>atlongragged4colborder (style)</code>	183	<code>descriptionpluralaccess (key)</code>	215
<code>atlongragged4colheader (style)</code>	182	<code>dua (option)</code>	14
<code>atlongragged4colheaderborder</code> <code>(style)</code>	183	<code>\DUANewAcronymDef</code>	154
<code>\altnewglossary</code>	40	E	
<code>altsuper4col (style)</code>	192	<code>entrycounter (option)</code>	7
<code>altsuper4colborder (style)</code>	193	<code>entrycounterwithin (option)</code>	7
<code>altsuper4colheader (style)</code>	193	<code>\entryname</code>	18
<code>altsuper4colheaderborder (style)</code>	193	<code>environments:</code>	
<code>altsuperragged4col (style)</code>	197	<code>align</code>	58
<code>altsuperragged4colborder (style)</code>	198	<code>description</code>	170
<code>altsuperragged4colheader (style)</code>	198	<code>longtable</code>	6 , 159 , 173–183
<code>altsuperragged4colheaderborder</code> <code>(style)</code>	199	<code>multicols</code>	184
<code>almtree (style)</code>	204	<code>supertabular</code>	6 , 159 , 187–199
<code>almtreegroup (style)</code>	206	<code>theglossary</code>	23 , 133 , 134 , 136 , 185 , 186 , 201 , 202 , 204
<code>almtreehypergroup (style)</code>	207	<code>theindex</code>	199
<code>amsgen package</code>	3 , 56	<code>equation counter</code>	58 , 59
<code>\andname</code>	18	<code>etoolbox package</code>	3 , 164
<code>array package</code>	178 , 194	F	
<code>article class</code>	124	<code>file types</code>	
B		<code>.aux</code>	129
<code>babel package</code>	16 , 17 , 19 , 34 , 235	<code>.glo</code>	51
C		<code>.ist</code>	112 , 119
<code>\capitalisewords</code>	165	<code>.toc</code>	26
<code>compatible-2.07 (option)</code>	16	<code>.xdy</code>	21
<code>counter (key)</code>	43	<code>glo</code>	163
<code>counter (option)</code>	11	<code>\findrootlanguage</code>	33
<code>\CustomAcronymFields</code>	155	<code>first (key)</code>	42
<code>\CustomNewAcronymDef</code>	156	<code>firstaccess (key)</code>	214
D		<code>\firstacronymfont</code>	144
<code>\DeclareAcronymList</code>	10	<code>firstplural (key)</code>	42
<code>\DefaultNewAcronymDef</code> ...	145 , 231	<code>firstpluralaccess (key)</code>	215
<code>\defglstdisplay</code>	39 , 41 , 42 , 55 , 56	<code>footnote (option)</code>	14
<code>\defglstdisplayfirst</code>	39 , 41 , 42 , 55 , 56	<code>\FootnoteNewAcronymDef</code> ..	151 , 232
<code>\DefineAcronymSynonyms</code>	157	<code>\foralllglossaries</code>	36
		<code>\foralllglsentries</code>	36

<code>\forglsentries</code>	36	<code>user3</code>	44
G		<code>user4</code>	44
<code>\glolinkprefix</code>	58	<code>user5</code>	44
<code>glossareentry counter</code>	132	<code>user6</code>	44
<code>glossaries package</code>		<code>glossary package</code>	1 , 140
.....	35 , 113 , 158 , 170 , 207 , 214	<code>glossary styles:</code>	
<code>glossaries-accsupp package</code>	51 , 214	<code>altlist</code>	171 , 172
<code>\GlossariesWarning</code>	15	<code>altlistgroup</code>	171 , 172
<code>\GlossariesWarningNoLine</code>	16	<code>altlisthypergroup</code>	172
<code>\glossary</code>	39 , 119 , 122 , 136	<code>altlong4col</code>	177 , 178 , 182
<code>glossary counters:</code>		<code>altlong4colborder</code>	178
<code>glossaryentry</code>	131	<code>altlong4colheader</code>	178
<code>glossarysubentry</code>	131	<code>altlong4colheaderborder</code> ..	178
<code>glossary keys:</code>		<code>altlongragged4col</code>	182 , 183
<code>access</code>	214	<code>altlongragged4colborder</code> ..	183
<code>counter</code>	43	<code>altlongragged4colheader</code> ..	182
<code>description</code>	41	<code>altlongragged4colheaderborder</code>	183
<code>descriptionaccess</code>	215	183
<code>descriptionplural</code>	41	<code>altsuper4col</code>	192 , 193 , 197
<code>descriptionpluralaccess</code> ..	215	<code>altsuper4colborder</code>	193
<code>first</code>	42	<code>altsuper4colheader</code>	193
<code>firstaccess</code>	214	<code>altsuper4colheaderborder</code> .	193
<code>firstplural</code>	42	<code>altsuperragged4col</code> ...	197 – 199
<code>firstpluralaccess</code>	215	<code>altsuperragged4colborder</code> .	198
<code>long</code>	44	<code>altsuperragged4colheader</code> .	198
<code>longaccess</code>	215	<code>altsuperragged4colheaderborder</code>	199
<code>longplural</code>	44	199
<code>longpluralaccess</code>	215	<code>alttree</code>	186 , 204 , 206
<code>name</code>	41	<code>alttreegroup</code>	206 , 207
<code>nonumberlist</code>	43	<code>alttreehypergroup</code>	207
<code>parent</code>	43	<code>index</code>	184 , 199 – 201
<code>plural</code>	42	<code>indexgroup</code>	200 , 201
<code>pluralaccess</code>	215	<code>indexhypergroup</code>	201
<code>see</code>	43	<code>inline</code>	168
<code>short</code>	44	<code>list</code>	5 , 170 – 172
<code>shortaccess</code>	215	<code>listdotted</code>	172 , 173
<code>shortplural</code>	44	<code>listgroup</code>	171
<code>shortpluralaccess</code>	215	<code>listhypergroup</code>	171
<code>sort</code>	41	<code>long</code>	173 , 174 , 179
<code>symbol</code>	42	<code>long3col</code>	174 , 175
<code>symbolaccess</code>	215	<code>long3colborder</code>	175
<code>symbolplural</code>	42	<code>long3colheader</code>	175
<code>symbolpluralaccess</code>	215	<code>long3colheaderborder</code>	176
<code>text</code>	41	<code>long4col</code>	176 , 177
<code>textaccess</code>	214	<code>long4colborder</code>	177
<code>type</code>	42	<code>long4colheader</code>	176
<code>user1</code>	43	<code>long4colheaderborder</code>	177
<code>user2</code>	43	<code>longborder</code>	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	112
longragged3colheader	181	glossary-list package	5 , 6 , 170
longragged3colheaderborder	181	glossary-long package	6 , 173 , 182 , 187 , 188
longraggedborder	179	glossary-longragged package	178
longraggedheader	180	glossary-mcols package	184
longraggedheaderborder	180	glossary-super package	
mcolalttree	186	6 , 173 , 187 , 194 , 197
mcolalttreegroup	186 , 187	glossary-superragged package	194
mcolalttreehypergroup	187	glossary-tree package	6 , 199
mcolindex	184	glossaryentry (counter)	131
mcolindexgroup	184	glossaryentry counter	7 , 132 , 133
mcolindexhypergroup	184	\glossaryentryfield	42 , 134 , 136 , 137
mcoltree	185	\glossaryentrynumber	131
mcoltreegroup	185	\glossaryentrynumbers	5 , 22 , 128 , 129
mcoltreehypergroup	185	\glossaryheader	134 , 136
mcoltreenoname	185 , 186	\glossarymark	6 , 24
mcoltreenonamegroup	186	\glossaryname	17 , 19
mcoltreenonamehypergroup	186	\glossarypostamble	23 , 136
sublistdotted	172	\glossarypreamble	23 , 136
super	188 , 189 , 195	\glossarysection	4 , 23 , 39
super3col	189 , 190	\glossarystyle	136 , 159
super3colborder	190	glossarysubentry (counter)	131
super3colheader	190	glossarysubentry counter	7 , 132 , 133
super3colheaderborder	190	\GLS	70
super4col	191 , 192	\Gls	69 , 72 , 164
super4colborder	192	\gls	3 , 42 , 54 , 55 , 67 , 70 , 71 , 75 , 77 , 78 , 80 , 81 , 83 , 84 , 86 , 87 , 89 , 90 , 92 , 93 , 95 , 96 , 133
super4colheader	191	\gls@checkisacronymlist	11
super4colheaderborder	192	\gls@codepage	15
superborder	188	\gls@docclearpage	25
superheader	189	\gls@hypergroup prerun	167
superheaderborder	189	\gls@level	45
superragged	194 – 196	\gls@save@numberlist	127
superragged3col	196 , 197	\gls@suffixF	22
superragged3colborder	196	\gls@suffixFF	22
superragged3colheader	197	\glsaccessdisplay	221
superragged3colheaderborder	197	\glsaccsupp	218
superraggedborder	195	\glsadd	54 , 111 , 136
superraggedheader	195	\glsadd options	
superraggedheaderborder	195	counter	111
superraggedright3colheaderborder	197	format	111 , 137
tree	185 , 201 – 204	\glsaddall	54 , 112
treegroup	185 , 202	\glsaddall options	
		types	111 , 112

<code>\GlsAddLetterGroup</code>	36	<code>\Glsentryfullpl</code>	110
<code>\GlsAddSortRule</code>	33	<code>\glentryfullpl</code>	109
<code>\GlsAddXdyAlphabet</code>	29	<code>\glentryitem</code>	133
<code>\GlsAddXdyAttribute</code>	28, 207	<code>\Glsentrylong</code>	109
<code>\GlsAddXdyCounters</code>	27, 208	<code>\glentrylong</code>	109
<code>\GlsAddXdyLocation</code>	31, 208	<code>\glentrylongaccess</code>	218
<code>\GlsAddXdyStyle</code>	33	<code>\Glsentrylongpl</code>	109
<code>\glsautoprefix</code>	4	<code>\glentrylongpl</code>	109
<code>\glsclearpage</code>	26	<code>\glentrylongpluralaccess</code> ..	218
<code>\glsclosebrace</code>	112	<code>\Glsentryname</code>	105
<code>\glscompositor</code>	21, 31	<code>\glentryname</code>	105, 126
<code>\glscounter</code>	11, 40	<code>\glentrynumberlist</code>	110
<code>\glsdefaulttype</code>	9	<code>\Glsentryplural</code>	106
<code>\glsdefmain</code>	9	<code>\glentryplural</code>	106
<code>\GLSdesc</code>	84	<code>\glentrypluralaccess</code>	217
<code>\Glsdesc</code>	83	<code>\Glsentryshort</code>	109
<code>\glsdesc</code>	83, 84	<code>\glentryshort</code>	108
<code>\GLSdescplural</code>	85	<code>\glentryshortaccess</code>	218
<code>\Glsdescplural</code>	85	<code>\Glsentryshortpl</code>	109
<code>\glsdescplural</code>	84, 85	<code>\glentryshortpl</code>	109
<code>\glsdescriptionaccessdisplay</code>	220	<code>\glentryshortpluralaccess</code> ..	218
<code>\glsdescriptionpluralaccessdisplay</code>	220	<code>\glentrysort</code>	107
<code>\glsdescwidth</code>	173, 179, 187, 194	<code>\Glsentrysymbol</code>	106
<code>\glsdisablehyper</code>	67	<code>\glentrysymbol</code>	106
<code>\glsdisp</code>	74	<code>\glentrysymbolaccess</code>	218
<code>\glsdisplay</code>	12, 41, 42, 54, 55, 67	<code>\Glsentrysymbolplural</code>	107
<code>\glsdisplayfirst</code> ..	41, 42, 54, 55, 67	<code>\glentrysymbolplural</code>	106
<code>\glsdisplaynumberlist</code>	110	<code>\glentrysymbolpluralaccess</code> ..	218
<code>\glsdoifexists</code>	37	<code>\Glsentrytext</code>	106
<code>\glsdoifnoexists</code>	37	<code>\glentrytext</code>	106, 126
<code>\glsenablehyper</code>	67	<code>\glentrytextaccess</code>	217
<code>\glentryaccess</code>	217	<code>\glentrytype</code>	107
<code>\glentrycounter</code>	58	<code>\Glsentryuseri</code>	107
<code>\glentrycounterlabel</code>	133	<code>\glentryuseri</code>	107
<code>\Glsentrydesc</code>	105	<code>\Glsentryuserii</code>	108
<code>\glentrydesc</code>	12, 105	<code>\glentryuserii</code>	108
<code>\glentrydescaccess</code>	218	<code>\glentryuseriii</code>	108
<code>\Glsentrydescplural</code>	106	<code>\glentryuseriii</code>	108
<code>\glentrydescplural</code>	106	<code>\Glsentryuseriv</code>	108
<code>\glentrydescpluralaccess</code> ..	218	<code>\glentryuseriv</code>	108
<code>\Glsentryfirst</code>	107	<code>\Glsentryuserv</code>	108
<code>\glentryfirst</code>	107	<code>\glentryuserv</code>	108
<code>\glentryfirstaccess</code>	217	<code>\glentryuservi</code>	108
<code>\Glsentryfirstplural</code>	107	<code>\GLSfirst</code>	78
<code>\glentryfirstplural</code>	107	<code>\Glsfirst</code>	77, 78
<code>\glentryfirstpluralaccess</code> ..	217	<code>\glsfirst</code>	77
<code>\Glsentryfull</code>	109	<code>\glsfirstaccessdisplay</code>	219
<code>\glentryfull</code>	109	<code>\GLSfirstplural</code>	81

<code>\Glsfirstplural</code>	80	<code>\glsnumbersgroupname</code> ..	18 , 112 , 135
<code>\glsfirstplural</code>	80 , 81	<code>\glsnumlistlastsep</code>	111
<code>\glsfirstpluralaccessdisplay</code>	220	<code>\glsnumlistsep</code>	111
<code>\glsgetgrouplabel</code>	135	<code>\glsopenbrace</code>	112
<code>\glsgetgrouptitle</code>	112 , 135	<code>\glsorder</code>	14
<code>\glsgroupheading</code>	135 , 136	<code>\glspagelistwidth</code>	173 , 179 , 188 , 194
<code>\glsgroupskip</code>	135 , 136 , 170	<code>\glspar</code>	20
<code>\glshyperlink</code>	111	<code>\GLSpl</code>	73
<code>\glshypernavsep</code>	168	<code>\Glspl</code>	72 , 164
<code>\glshypernumber</code>	22 , 137	<code>\glspl</code>	54 , 55 , 71 – 73
<code>\glsIfListOfAcronyms</code>	10	<code>\GLSplural</code>	79
<code>\glsinlineparentchildseparator</code>	170	<code>\Glsplural</code>	79
<code>\glsinlineseparator</code>	169	<code>\glsplural</code>	78 , 79
<code>\glsinlinesubseparator</code>	170	<code>\glspluralaccessdisplay</code>	219
<code>\glskeylisttok</code>	144	<code>\glspluralsuffix</code>	18 , 42
<code>\glslabeltok</code>	144	<code>\glspostdescription</code>	20
<code>\glslink</code>	54 , 57 , 67 , 111 , 136 , 137	<code>\glspostinline</code>	170
<code>\glslink options</code>		<code>\glsquote</code>	113
counter	56 , 67 , 164	<code>\glsrefentry</code>	133
format	56 , 67 , 137	<code>\glsreset</code>	53
hyper	56 , 67	<code>\glsresetall</code>	53
<code>\glslistdottedwidth</code>	172	<code>\glsresetentrylist</code>	131
<code>\glslocalreset</code>	53	<code>\glsresetsubentrycounter</code>	132
<code>\glslocalresetall</code>	53	<code>\glssee</code>	126
<code>\glslocalunset</code>	53	<code>\glsseeformat</code>	115 , 126
<code>\glslocalunsetall</code>	54	<code>\glsseeitem</code>	126
<code>\glslongaccessdisplay</code>	221	<code>\glsseeitemformat</code>	126
<code>\glslongaccesskey</code>	234	<code>\glsseelastsep</code>	126
<code>\glslongkey</code>	141	<code>\glsseelist</code>	126
<code>\glslongpluralaccessdisplay</code> .	221	<code>\glsseesep</code>	126
<code>\glslongpluralaccesskey</code>	234	<code>\glsSetAlphaCompositor</code>	22
<code>\glslongpluralkey</code>	141	<code>\glsSetCompositor</code>	21
<code>\glslongtok</code>	144	<code>\glsSetSuffixF</code>	22
<code>\glsmakefirsttuc</code>	165	<code>\glsSetSuffixFF</code>	22
<code>\glsmcols</code>	184	<code>\glssettoctitle</code>	18
<code>\glsmoveentry</code>	50	<code>\glssetwidest</code>	204
<code>\GLSname</code>	82	<code>\GlsSetXdyCodePage</code>	35
<code>\Glsname</code>	82	<code>\GlsSetXdyFirstLetterAfterDigits</code>	113
<code>\glsname</code>	81 , 82	<code>\GlsSetXdyLanguage</code>	35
<code>\glsnameaccessdisplay</code>	219	<code>\GlsSetXdyLocationClassOrder</code> .	32
<code>\glsnamefont</code>	137	<code>\GlsSetXdyMinRangeLength</code>	113
<code>\glsnavhyperlink</code>	166	<code>\GlsSetXdyStyles</code>	33
<code>\glsnavhypertarget</code>	167	<code>\glsshortaccessdisplay</code>	221
<code>\glsnavigation</code>	168	<code>\glsshortaccesskey</code>	234
<code>\glsnextpages</code>	131	<code>\glsshortkey</code>	141
<code>\glsnonextpages</code>	131	<code>\glsshortpluralaccessdisplay</code>	221
<code>\glsnoxindywarning</code>	26	<code>\glsshortpluralaccesskey</code>	234
<code>\glsnumberformat</code>	22	<code>\glsshortpluralkey</code>	141

long4colheaderborder (style) ...	177	multicol package	184
longaccess (key)	215	multicols (environment)	184
longborder (style)	174		
longheader (style)	174		
longheaderborder (style)	174		
longplural (key)	44		
longpluralaccess (key)	215		
longragged (style)	179		
longragged3col (style)	180		
longragged3colborder (style) ...	181		
longragged3colheader (style) ...	181		
longragged3colheaderborder (style)	181		
longraggedborder (style)	179		
longraggedheader (style)	180		
longraggedheaderborder (style) .	180		
longtable (environment)	6 , 159 , 173–183		
longtable package	173 , 179		
	M		
\makefirstuc	164		
makeglossaries	14 , 21 , 33 , 35 , 39 , 120 , 127 , 129		
\makeglossaries	17 , 21 , 22 , 38 , 40 , 120 , 121		
\makeglossary	121		
makeindex	12 , 14 , 15 , 18 , 21 , 23 , 38–41 , 52 , 59 , 62 , 112 , 115 , 117 , 119 , 124 , 134 , 135 , 209		
delim_n	23		
delim_r	23		
page_compositor	21		
special characters	60 , 61 , 112		
\MakeUppercase	6		
mcolalttree (style)	186		
mcolalttreegroup (style)	186		
mcolalttreehypergroup (style) ..	187		
mcolindex (style)	184		
mcolindexgroup (style)	184		
mcolindexhypergroup (style)	184		
mcoltree (style)	185		
mcoltreegroup (style)	185		
mcoltreehypergroup (style)	185		
mcoltreenoname (style)	185		
mcoltreenonamegroup (style)	186		
mcoltreenonamehypergroup (style)	186		
memoir class	122		
mfirstuc package	1		
		N	
		name (key)	41
		\newacronym	14 , 44 , 54 , 140 , 141
		\newacronymhook	144
		\newglossary .	11 , 39 , 40 , 119 , 120 , 130
		\newglossaryentry	12 , 45 , 54 , 140 , 141
		\newglossaryentry options	
		access	216 , 217
		counter	43
		description	12 , 14 , 41 , 45 , 55 , 83 , 105 , 141 , 151 , 215
		descriptionaccess	218 , 220
		descriptionplural	84 , 215
		descriptionpluralaccess ...	218 , 220
		first	42 , 48 , 55 , 67 , 77 , 107 , 150 , 153 , 214
		firstaccess	217 , 219
		firstplural	42 , 48 , 55 , 80 , 107 , 215
		firstpluralaccess	217 , 220
		format	114
		long	109 , 215
		longaccess	218 , 221
		longplural	109 , 215
		longpluralaccess	218 , 221
		name	12 , 41 , 44 , 45 , 81 , 105 , 126 , 214
		nonumberlist	43
		parent	43 , 45
		plural	42 , 48 , 55 , 78 , 215
		pluralaccess	217 , 219
		see	5 , 43
		short	108 , 215
		shortaccess	218 , 221
		shortplural	109 , 215
		shortpluralaccess	218 , 221
		sort	12 , 41 , 107 , 134 , 135
		symbol	12 , 13 , 41 , 42 , 55 , 86 , 106 , 147 , 148 , 150 , 153 , 176 , 191 , 214–216
		symbolaccess	218 , 220
		symbolplural	87 , 215
		symbolpluralaccess	218 , 220
		text	41 , 42 , 55 , 67 , 75 , 106 , 147 , 150 , 214
		textaccess	217 , 219
		type	9 , 42 , 54 , 107
		user1	89 , 107 , 216

user2	90, 108	nolong	159, 173
user3	92, 108	nolong	6
user4	93, 108	nomain	10
user5	95, 108	nonumberlist	5
user6	96, 108, 216	nonumberlist	5
\newglossarystyle	136	nostyles	6
ngerman package	34	nosuper	159
\noist	119, 163, 214	nosuper	6
nolist (option)	6	notree	159
nolong (option)	6	notree	6
nonumberlist (key)	43	numberedsection	4
nonumberlist (option)	5	numberline	4
\nopostdesc	20	numberline	4
nostyles (option)	6	order	14
nosuper (option)	6	sanitize	12, 13, 41, 105, 106
notree (option)	6	sanitize	13
numberedsection (option)	4	savenumberlist	5
numberline (option)	4	savewrites	15
O		false	119
\oldacronym	140	true	121
order (option)	14	savewrites	15
P		section	4, 25
package options:		section	4
acronym	10, 17, 129, 140, 141	seeautonumberlist	5
acronym	10	shotcuts	14
acronymlists	11	smallcaps	14
compatible-2.07	16	smaller	14
counter	11	sort	
counter	11	def	7, 8
description	150	standard	7
description	14	use	7, 8
dua	149, 150	sort	7
dua	14	style	5, 158, 159
entrycounter	131	style	5
true	7	subentrycounter	131
entrycounter	7	subentrycounter	7
entrycounterwithin	7	toc	4
footnote	68–71,	true	4
73–75, 147, 149–151, 222–227		toc	4
footnote	14	translate	13
hyperfirst		translate	13
false	68–71, 73–75	ucmark	6
hyperfirst	13	xindy	15, 115, 164
indexonlyfirst	13	\pagelistname	18
makeindex	115, 164	parent (key)	43
nolist	158	\phantomsection	24, 25
nolist	6	plural (key)	42
		pluralaccess (key)	215
		polyglossia package	16, 19

<code>\printglossaries</code>	<code>\setglossarysection</code>
... 9 , 23 , 38 , 40 , 121 , 127 , 130 , 166	<code>\SetSmallAcronymDisplayStyle</code> 152
<code>\printglossary</code>	<code>\SetSmallAcronymStyle</code>
... 23 , 38 , 121 , 127 , 129 , 130 , 166	<code>\setStyleFile</code>
<code>\printglossary options</code>	<code>short (key)</code>
<code>nonumberlist</code>	<code>shortaccess (key)</code>
<code>numberedsection</code>	<code>shortplural (key)</code>
<code>style</code>	<code>shortpluralaccess (key)</code>
<code>title</code>	<code>shotcuts (option)</code>
<code>toctitle</code>	<code>\showacronymlists</code>
<code>type</code>	<code>\showglocounter</code>
<code>\protect</code>	<code>\showglodesc</code>
	<code>\showglodescaccess</code>
R	<code>\showglodescplural</code>
<code>\renewglossarystyle</code>	<code>\showglodescpluralaccess</code>
<code>\roman</code>	<code>\showglofirst</code>
<code>\rootlanguage</code>	<code>\showglofirstaccess</code>
	<code>\showglofirststpl</code>
S	<code>\showglofirststplpluralaccess</code>
<code>sanitize (option)</code>	<code>\showgloflag</code>
<code>savenumberlist (option)</code>	<code>\showgloindex</code>
<code>savewrites (option)</code>	<code>\showglolevel</code>
<code>section (option)</code>	<code>\showglolongaccess</code>
<code>see (key)</code>	<code>\showglolongpluralaccess</code>
<code>seeautonumberlist (option)</code>	<code>\showgloname</code>
<code>\seename</code>	<code>\showglonameaccess</code>
<code>\SetAcronymLists</code>	<code>\showgloparent</code>
<code>\SetAcronymStyle</code>	<code>\showgloplural</code>
<code>\SetCustomDisplayStyle</code>	<code>\showglopluralaccess</code>
<code>\SetCustomStyle</code>	<code>\showgloshortaccess</code>
<code>\SetDefaultAcronymDisplayStyle</code>	<code>\showgloshortpluralaccess</code>
.....	<code>\showglosort</code>
<code>\SetDefaultAcronymStyle</code>	<code>\showglossaries</code>
<code>\SetDescriptionAcronymDisplayStyle</code>	<code>\showglossarycounter</code>
.....	<code>\showglossaryentries</code>
<code>\SetDescriptionAcronymStyle</code>	<code>\showglossaryin</code>
<code>\SetDescriptionDUAAcronymDisplayStyle</code>	<code>\showglossaryout</code>
.....	<code>\showglossarytitle</code>
<code>\SetDescriptionDUAAcronymStyle</code>	<code>\showglosymbol</code>
.....	<code>\showglosymbolaccess</code>
<code>\SetDescriptionFootnoteAcronymDisplayStyle</code>	<code>\showglosymbolplural</code>
.....	<code>\showglosymbolpluralaccess</code>
<code>\SetDescriptionFootnoteAcronymStyle</code>	<code>\showglotext</code>
.....	<code>\showglotextaccess</code>
<code>\SetDUADisplayStyle</code>	<code>\showglotype</code>
<code>\SetDUASStyle</code>	<code>\showglouser</code>
<code>\setentrycounter</code>	<code>\showglouserii</code>
<code>\SetFootnoteAcronymDisplayStyle</code>	<code>\showglouseriii</code>
.....	<code>\showglouseriv</code>
<code>\SetFootnoteAcronymStyle</code>	

