

Documented Code For glossaries v3.01

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/>

2011-04-12

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

`glossariesbegin.pdf` If you are a complete beginner, start with “The glossaries package: a guide for beginners”.

`glossary2glossaries.pdf` If you are moving over from the obsolete `glossary` package, read “Upgrading from the `glossary` package to the `glossaries` package”.

`glossaries-user.pdf` For the main user guide, read “`glossaries.sty` v3.01: L^AT_EX₂ ϵ 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 | 3 |
| 1.3 | Default values | 16 |
| 1.4 | Xindy | 24 |
| 1.5 | Loops and conditionals | 34 |
| 1.6 | Defining new glossaries | 36 |
| 1.7 | Defining new entries | 38 |
| 1.8 | Resetting and unsetting entry flags | 48 |
| 1.9 | Loading files containing glossary entries | 50 |
| 1.10 | Using glossary entries in the text | 50 |
| 1.10.1 | Links to glossary entries | 51 |
| 1.10.2 | Displaying entry details without adding information to the glossary | 100 |
| 1.11 | Adding an entry to the glossary without generating text | 105 |
| 1.12 | Creating associated files | 106 |
| 1.13 | Writing information to associated files | 115 |
| 1.14 | Glossary Entry Cross-References | 118 |
| 1.15 | Displaying the glossary | 120 |
| 1.16 | Acronyms | 131 |
| 1.17 | Predefined acronym styles | 135 |
| 1.18 | Predefined Glossary Styles | 150 |
| 1.19 | Debugging Commands | 150 |
| 1.20 | Compatibility with version 2.07 and below | 155 |
| 2 | Mfirstuc Documented Code | 155 |
| 3 | Glossary Styles | 157 |
| 3.1 | Glossary hyper-navigation definitions (glossary-hypernav package) | 157 |
| 3.2 | List Style (glossary-list.sty) | 159 |
| 3.3 | Glossary Styles using longtable (the glossary-long package) | 162 |
| 3.4 | Glossary Styles using longtable (the glossary-longragged package) | 168 |
| 3.5 | Glossary Styles using supertabular environment (glossary-super package) | 173 |
| 3.6 | Glossary Styles using supertabular environment (glossary-superragged package) | 179 |
| 3.7 | Tree Styles (glossary-tree.sty) | 185 |
| 4 | glossaries-compatible-207 | 192 |
| 5 | Accessibility Support (glossaries-accsupp Code) | 199 |
| 5.1 | Defining Replacement Text | 199 |
| 5.2 | Accessing Replacement Text | 202 |
| 5.3 | Displaying the Glossary | 215 |

| | | |
|----------|------------------------------|------------|
| 5.4 | Acronyms | 215 |
| 5.5 | Debugging Commands | 219 |
| 6 | Multi-Lingual Support | 220 |
| 6.1 | Babel Captions | 220 |
| 6.2 | Polyglossia Captions | 226 |
| 6.3 | Brazilian Dictionary | 229 |
| 6.4 | Danish Dictionary | 230 |
| 6.5 | Dutch Dictionary | 230 |
| 6.6 | English Dictionary | 230 |
| 6.7 | French Dictionary | 230 |
| 6.8 | German Dictionary | 231 |
| 6.9 | Irish Dictionary | 231 |
| 6.10 | Italian Dictionary | 231 |
| 6.11 | Magyar Dictionary | 232 |
| 6.12 | Polish Dictionary | 232 |
| 6.13 | Serbian Dictionary | 232 |
| 6.14 | Spanish Dictionary | 233 |
| | Index | 234 |

1 Main Package Code

1.1 Package Definition

This package requires L^AT_EX 2 ϵ .

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries}[2011/04/12 v3.01 (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}
```

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]{}}
```

`\glo@seeautonumberlist`

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

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

```
41 \DeclareOptionX{seeautonumberlist}{%
```

```
42 \renewcommand*\glo@seeautonumberlist{%
```

```
43 \def\glo@prefix{glsnextpages}%
```

```
44 }%
```

```
45 }
```

`\gls@loadlong`

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

nolong This option prevents from being loaded. This means that the glossary styles that use the `longtable` environment will not be available. This option is provided to reduce overhead caused by loading unrequired packages.

```
47 \DeclareOptionX{nolong}{\renewcommand*\gls@loadlong{}}
```

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

```
48 \IfFileExists{supertabular.sty}{%
```

```
49 \newcommand*\gls@loadsuper{\RequirePackage{glossary-super}}{%
```

```
50 \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.

```
51 \DeclareOptionX{nosuper}{\renewcommand*{\@gls@loadsuper}{}}
```

`\@gls@loadlist`

```
52 \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.

```
53 \DeclareOptionX{nolist}{\renewcommand*{\@gls@loadlist}{}}
```

`\@gls@loadtree`

```
54 \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.

```
55 \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).

```
56 \DeclareOptionX{nostyles}{%
57 \renewcommand*{\@gls@loadlong}{}%
58 \renewcommand*{\@gls@loadsuper}{}%
59 \renewcommand*{\@gls@loadlist}{}%
60 \renewcommand*{\@gls@loadtree}{}%
61 \let\@glossary@default@style\relax
62 }
```

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`.

```
63 \define@boolkey{glossaries.sty}[gls]{entrycounter}[true]{}
64 \glseentrycounterfalse
```

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

```
65 \define@key{glossaries.sty}{counterwithin}{%
66 \renewcommand*{\@gls@counterwithin}{#1}%
67 \glseentrycountertrue
68 }
```

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

```
69 \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`.

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

```

sort Define the sort method: sort=standard (default), sort=def (order of definition) or
sort=use (order of use).
72 \define@choicekey{glossaries.sty}{sort}{standard,def,use}{%
73 \csname @gls@setupsort@#1\endcsname
74 }

\gls@setupsort@standard Set up the macros for default sorting.
75 \newcommand*{\@gls@setupsort@standard}{%
Store entry information when it's defined.
76 \def\do@glo@storeentry{\@glo@storeentry}%
No count register required for standard sort.
77 \def\@gls@defsortcount##1{%
Sort according to sort key (\@glo@sort) if provided otherwise sort according to
the entry's name (\@glo@name).
78 \def\@gls@defsort##1##2{%
79 \ifx\@glo@sort\@glsdefaultsort
80 \let\@glo@sort\@glo@name
81 \fi
82 \@onelevel@sanitize\@glo@sort
83 \expandafter\protected@xdef\csname glo@##2@sort\endcsname{\@glo@sort}%
84 }%
Don't need to do anything when the entry is used.
85 \def\@gls@setsort##1{%
86 }
Set standard sort as the default:
87 \@gls@setupsort@standard

\glssortnumberfmt Format the number used as the sort key by sort=def and sort=use. Defaults to six
digit numbering.
88 \newcommand*\glssortnumberfmt[1]{%
89 \ifnum#1<100000 0\fi
90 \ifnum#1<10000 0\fi
91 \ifnum#1<1000 0\fi
92 \ifnum#1<100 0\fi
93 \ifnum#1<10 0\fi
94 \number#1%
95 }

\@gls@setupsort@def Set up the macros for order of definition sorting.
96 \newcommand*{\@gls@setupsort@def}{%
Store entry information when it's defined.
97 \def\do@glo@storeentry{\@glo@storeentry}%

```

Defined count register associated with the glossary.

```
98 \def\@gls@defsortcount##1{%
99   \expandafter\global
100   \expandafter\newcount\csname glossary@##1@sortcount\endcsname
101 }%
```

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

```
102 \def\@gls@defsort##1##2{%
103   \expandafter\global\expandafter
104   \advance\csname glossary@##1@sortcount\endcsname by 1\relax
105   \expandafter\protected\xdef\csname glo@##2@sort\endcsname{%
106     \expandafter\glssortnumberfmt
107     {\csname glossary@##1@sortcount\endcsname}}%
108 }%
```

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

```
109 \def\@gls@setsort##1{%
110 }
```

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

```
111 \newcommand*\@gls@setupsort@use{%
```

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

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

Defined count register associated with the glossary.

```
113 \def\@gls@defsortcount##1{%
114   \expandafter\global
115   \expandafter\newcount\csname glossary@##1@sortcount\endcsname
116 }%
```

Initialise the sort key to empty.

```
117 \def\@gls@defsort##1##2{%
118   \expandafter\gdef\csname glo@##2@sort\endcsname{}%
119 }%
```

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

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

Get the parent, if one exists

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

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

```
122   \ifx\@glo@parent\@empty
```

```
123   \else
```

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

```
125   \fi
```

Set index information for this entry

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

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

```
128   \ifx\@gls@tmp\@empty
```

```

129     \expandafter\global\expandafter
130     \advance\csname glossary@\@glo@type @sortcount\endcsname by 1\relax
131     \expandafter\protected\edef\csname glo@##1@sort\endcsname{%
132         \expandafter\glssortnumberfmt
133         {\csname glossary@\@glo@type @sortcount\endcsname}}%
134     \@glo@storeentry{##1}%
135     \fi
136 }%
137 }

```

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

```

138 \newcommand*\glsdefmain}{%
139   \newglossary{main}{gls}{glo}{\glossaryname}%
140 }

```

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

```

141 \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

```

142 \newcommand*\acronymtype}{\glsdefaulttype}

```

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

```

143 \DeclareOptionX{nomain}{%
144   \let\glsdefaulttype\relax
145   \renewcommand*\glsdefmain}{}%
146 }

```

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.

```

147 \define@boolkey{glossaries.sty}{gls}{acronym}[true]{%
148   \DeclareAcronymList{acronym}%
149 }

```

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

```

150 \newcommand*\@glsacronymlists}{

```

`\@addtoacronymlists`

```
151 \newcommand*\@addtoacronymlists}[1]{%
152   \ifx\@glsacronymlists\@empty
153     \protected@xdef\@glsacronymlists{#1}%
154   \else
155     \protected@xdef\@glsacronymlists{\@glsacronymlists,#1}%
156   \fi
157 }
```

`\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.)

```
158 \newcommand*\DeclareAcronymList}[1]{%
159   \glsIfListOfAcronyms{#1}{\@addtoacronymlists{#1}}%
160 }
```

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

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

```
161 \newcommand*\glsIfListOfAcronyms}[1]{%
162   \edef\@do@gls@islistofacronyms{%
163     \noexpand\@gls@islistofacronyms{#1}{\@glsacronymlists}}%
164   \@do@gls@islistofacronyms
165 }
```

Internal command requires label and list to be expanded:

```
166 \newcommand*\@gls@islistofacronyms}[4]{%
167   \def\gls@islistofacronyms##1,#1,##2\end@gls@islistofacronyms{%
168     \def\@before{##1}\def\@after{##2}}%
169   \gls@islistofacronyms,#2,#1,\@nil\end@gls@islistofacronyms
170   \ifx\@after\@nnil
```

Not found

```
171   #4%
172   \else
```

Found

```
173   #3%
174   \fi
175 }
```

`\if@glsisacronymlist` Convenient boolean.

```
176 \newif\if@glsisacronymlist
```

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

```
177 \newcommand*\gls@checkisacronymlist}[1]{%
178   \glsIfListOfAcronyms{#1}%
179   {\@glsisacronymlisttrue}{\@glsisacronymlistfalse}%
180 }
```

`\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.)

```
181 \newcommand*\SetAcronymLists[1]{%
182 \renewcommand*\@glsacronymlists{#1}%
183 }
```

`acronymlists`

```
184 \define@key{glossaries.sty}{acronymlists}{%
185 \addtoacronymlists{#1}%
186 }
```

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`

```
187 \newcommand{\glscounter}{page}
```

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

```
188 \define@key{glossaries.sty}{counter}{%
189 \renewcommand*\glscounter{#1}%
190 }
```

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`

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

`\@gls@sanitizename`

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

`\@gls@sanitizesymbol`

```
193 \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.

```
194 \define@boolkey[gls]{sanitize}{description}[true]{%
195 \ifgls@sanitize@description
196 \renewcommand*{\@gls@sanitizedesc}{\@onelevel@sanitize\@glo@desc}%
197 \else
198 \renewcommand*{\@gls@sanitizedesc}{}%
199 \fi
200 }
```

Similarly for the name key:

```
201 \define@boolkey[gls]{sanitize}{name}[true]{%
202 \ifgls@sanitize@name
203 \renewcommand*{\@gls@sanitizename}{\@onelevel@sanitize\@glo@name}%
204 \else
205 \renewcommand*{\@gls@sanitizename}{}%
206 \fi}
```

and for the symbol key:

```
207 \define@boolkey[gls]{sanitize}{symbol}[true]{%
208 \ifgls@sanitize@symbol
209 \renewcommand*{\@gls@sanitizesymbol}{%
210 \@onelevel@sanitize\@glo@symbol}%
211 \else
212 \renewcommand*{\@gls@sanitizesymbol}{}%
213 \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`:

```
214 \define@key{glossaries.sty}{sanitize}[description=true,symbol=true,
215 name=true]{%
216 \ifthenelse{\equal{#1}{none}}{%
217 \renewcommand*{\@gls@sanitizedesc}{}%
218 \renewcommand*{\@gls@sanitizename}{}%
219 \renewcommand*{\@gls@sanitizesymbol}{}%
220 }{\setkeys[gls]{sanitize}{#1}}%
221 }
```

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

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

Set the default value:

```

223 \glstranslatefalse
224 \ifpackageloaded{translator}{\glstranslatetrue}{%
225 \ifpackageloaded{babel}{\glstranslatetrue}{%
226 \ifpackageloaded{polyglossia}{\glstranslatetrue}{}}

```

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

```

227 \define@boolkey{glossaries.sty}[gls]{hyperfirst}[true]{
228 \glshyperfirsttrue

```

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

```

229 \define@boolkey{glossaries.sty}[glsacr]{footnote}[true]{%
230 \ifthenelse{\boolean{glsacrdescription}}{}}%
231 {\renewcommand*{\@gls@sanitizedesc}{}}%
232 }

```

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

```

233 \define@boolkey{glossaries.sty}[glsacr]{description}[true]{%
234 \renewcommand*{\@gls@sanitizesymbol}{}}%
235 }

```

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

```

236 \define@boolkey{glossaries.sty}[glsacr]{smallcaps}[true]{%
237 \renewcommand*{\@gls@sanitizesymbol}{}}%
238 }

```

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

```

239 \define@boolkey{glossaries.sty}[glsacr]{smaller}[true]{%
240 \renewcommand*{\@gls@sanitizesymbol}{}}%
241 }

```

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

```

242 \define@boolkey{glossaries.sty}[glsacr]{dua}[true]{%
243 \renewcommand*{\@gls@sanitizesymbol}{}}%
244 }

```

shortcuts Define acronym shortcuts.

```

245 \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.

```

246 \newcommand*{\glsorder}{word}

```

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

```

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

```

```

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

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

The default is makeindex:
251 \glsxindyfalse
Define package option to specify that makeindex will be used to sort the glos-
saries:
252 \DeclareOptionX{makeindex}{\glsxindyfalse}
The xindy package option may have a value which in turn can be a key=value
list. First define the keys for this sub-list. The boolean glsnumbers determines
whether to automatically add the glsnumbers letter group.
253 \define@boolkey[gls]{xindy}{glsnumbers}[true]{ }
254 \gls@xindy@glsnumberstrue

@xdy@main@language Define what language to use for each glossary type (if a language is not defined for
a particular glossary type the language specified for the main glossary is used.)
255 \def\@xdy@main@language{\rootlanguagename}%

Define key to set the language
256 \define@key[gls]{xindy}{language}{\def\@xdy@main@language{#1}}

\gls@codepage Define the code page. If \inputencodingname is defined use that, otherwise have
initialise with no codepage.
257 \ifcsundef{inputencodingname}{%
258 \def\gls@codepage{}}{%
259 \def\gls@codepage{\inputencodingname}
260 }

Define a key to set the code page.
261 \define@key[gls]{xindy}{codepage}{\def\gls@codepage{#1}}

Define package option to specify that xindy will be used to sort the glossaries:
262 \define@key{glossaries.sty}{xindy}[]{%
263 \glsxindytrue
264 \setkeys[gls]{xindy}{#1}%
265 }

savewrites The savewrites package option is provided to save on the number of write registers.
266 \define@boolkey{glossaries.sty}[gls]{savewrites}[true]{ }

Set default:
267 \glssavewritesfalse

```

`\GlossariesWarning` Prints a warning message.

```
268 \newcommand*\GlossariesWarning}[1]{%
269   \PackageWarning{glossaries}{#1}%
270 }
```

`\GlossariesWarningNoLine` Prints a warning message without the line number.

```
271 \newcommand*\GlossariesWarningNoLine}[1]{%
272   \PackageWarningNoLine{glossaries}{#1}%
273 }
```

Define package option to suppress warnings

```
274 \DeclareOptionX{nowarn}{%
275   \renewcommand*\GlossariesWarning}[1]{}%
276   \renewcommand*\GlossariesWarningNoLine}[1]{}%
277 }
```

compatible-2.07

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

Process package options:

```
280 \ProcessOptionsX
```

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

```
281 \ifglstranslate
282   \ifpackageloaded{babel}{\IfFileExists{translator.sty}{%
283     \RequirePackage{translator}}{}}{ }
284 \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}`).

```
285 \ifthenelse{\equal{\glscounter}{section}}{
286   {%
287     \ifcsundef{chapter}{%
288       {%
289         \let\gls@old@chapter\@chapter
290         \def\@chapter[#1]#2{\@gls@old@chapter[#1]#2}%
291         \ifcsundef{hyperdef}{\hyperdef{section}{\thesection}}{}}%
292       }%
293     }%
294 }
```

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

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

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

```
296 \newcommand*{\@onlypremakeg}[1]{%
297 \ifx\@gls@onlypremakeg\@empty
298   \def\@gls@onlypremakeg{#1}%
299 \else
300   \expandafter\toks@\expandafter{\@gls@onlypremakeg}%
301   \edef\@gls@onlypremakeg{\the\toks@,\noexpand#1}%
302 \fi}
```

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

```
303 \newcommand*{\@disable@onlypremakeg}{%
304 \@for\@thiscs:=\@gls@onlypremakeg\do{%
305   \expandafter\@disable@premakecs\@thiscs%
306 }}
```

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

```
307 \newcommand*{\@disable@premakecs}[1]{%
308   \def#1{\PackageError{glossaries}{\string#1\space may only be
309   used before \string\makeglossaries}{You can't use
310   \string#1\space after \string\makeglossaries}}%
311 }
```

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`

```
312 \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`

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

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

```
314 \newcommand*\glssettoctitle}[1]{%
315 \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
316 \providecommand*\entryname}{Notation}

\descriptionname
317 \providecommand*\descriptionname}{Description}

\symbolname
318 \providecommand*\symbolname}{Symbol}

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

Labels for makeindex's symbol and number groups:
\glssymbolsgroupname
320 \providecommand*\glssymbolsgroupname}{Symbols}

\glsnumbersgroupname
321 \providecommand*\glsnumbersgroupname}{Numbers}

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

\seename
323 \providecommand*\seename}{see}

\andname
324 \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.)
325 \newcommand*\addglossarytocaptions}[1]{%
326   \ifcsundef{captions#1}{}%
327   {%
328     \expandafter\let\expandafter\@gls@tmp\csname captions#1\endcsname
329     \expandafter\toks@\expandafter{\@gls@tmp
330       \renewcommand*\glossaryname}{\translate{Glossary}}}%
331     }%
332     \expandafter\edef\csname captions#1\endcsname{\the\toks@}%
333   }%
334 }

```

335 \ifglstranslate

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

```
336 \@ifpackageloaded{translator}{%
337   \usedictionary{glossaries-dictionary}%
338   \addglossarytocaptions{portuges}%
339   \addglossarytocaptions{portuguese}%
340   \addglossarytocaptions{brazil}%
341   \addglossarytocaptions{brazilian}%
342   \addglossarytocaptions{danish}%
343   \addglossarytocaptions{dutch}%
344   \addglossarytocaptions{afrikaans}%
345   \addglossarytocaptions{english}%
346   \addglossarytocaptions{UKenglish}%
347   \addglossarytocaptions{USenglish}%
348   \addglossarytocaptions{american}%
349   \addglossarytocaptions{australian}%
350   \addglossarytocaptions{british}%
351   \addglossarytocaptions{canadian}%
352   \addglossarytocaptions{newzealand}%
353   \addglossarytocaptions{french}%
354   \addglossarytocaptions{frenchb}%
355   \addglossarytocaptions{français}%
356   \addglossarytocaptions{acadian}%
357   \addglossarytocaptions{canadien}%
358   \addglossarytocaptions{german}%
359   \addglossarytocaptions{germanb}%
360   \addglossarytocaptions{austrian}%
361   \addglossarytocaptions{naustrian}%
362   \addglossarytocaptions{ngerman}%
363   \addglossarytocaptions{irish}%
364   \addglossarytocaptions{italian}%
365   \addglossarytocaptions{magyar}%
366   \addglossarytocaptions{hungarian}%
367   \addglossarytocaptions{polish}%
368   \addglossarytocaptions{spanish}%
369   \renewcommand*{\glstttitle}[1]{%
370     \ifthenelse{\equal{#1}{main}}{%
371       \translatelet{\glossarytoctitle}{Glossary}}{%
372       \ifthenelse{\equal{#1}{acronym}}{%
373         \translatelet{\glossarytoctitle}{Acronyms}}{%
374         \def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}}%
375   \renewcommand*{\glossaryname}{\translate{Glossary}}%
376   \renewcommand*{\acronymname}{\translate{Acronyms}}%
377   \renewcommand*{\entryname}{\translate{Notation (glossaries)}}%
378   \renewcommand*{\descriptionname}{%
379     \translate{Description (glossaries)}}%
380   \renewcommand*{\symbolname}{\translate{Symbol (glossaries)}}%
381   \renewcommand*{\pagelistname}{%
382     \translate{Page List (glossaries)}}%
```

```

383 \renewcommand*\glssymbolsgroupname}{%
384 \translate{Symbols (glossaries)}}%
385 \renewcommand*\glsnumbersgroupname}{%
386 \translate{Numbers (glossaries)}}%
387 }{%
388 \@ifpackageloaded{babel}%
389 {\RequirePackage{glossaries-babel}}%
390 {%
391 \@ifpackageloaded{polyglossia}{%
392 \RequirePackage{glossaries-polyglossia}}}%
393 }}
394 \fi

```

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

```
395 \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.

```
396 \newcommand*\nopostdesc}{}
```

`\@nopostdesc` Suppress next description terminator.

```

397 \newcommand*\@nopostdesc}{%
398 \let\org@glspostdescription\glspostdescription
399 \def\glspostdescription{%
400 \let\glspostdescription\org@glspostdescription}%
401 }

```

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

```
402 \newcommand*\glspar}{\par}
```

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

```

403 \ifglxindy
404 \newcommand*\setStyleFile}[1]{%
405 \renewcommand{\istfilename}{#1.xdy}}
406 \else
407 \newcommand*\setStyleFile}[1]{%
408 \renewcommand{\istfilename}{#1.ist}}
409 \fi

```

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

```
410 \@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`

```
411 \ifglxindy
412   \def\istfilename{\jobname.xdy}
413 \else
414   \def\istfilename{\jobname.ist}
415 \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 L^AT_EX, `\@istfilename` ignores its argument.

`\@istfilename`

```
416 \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`

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

`\glsSetCompositor` Sets the compositor.

```
418 \newcommand*\glsSetCompositor[1]{%
419   \renewcommand*\glscompositor{#1}}
```

Only use before `\makeglossaries`

```
420 \@onlypremakeg\glsSetCompositor
```

(The page compositor is usually defined as a dash when using `makeindex`, but most of the standard counters used by L^AT_EX 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.

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

`\glsSetAlphaCompositor` Sets the alpha compositor.

```
422 \ifglxindy
423   \newcommand*\glsSetAlphaCompositor[1]{%
424     \renewcommand*\@glsAlphacompositor{#1}}
425 \else
426   \newcommand*\glsSetAlphaCompositor[1]{%
427     \glsnoxindywarning\glsSetAlphaCompositor}
428 \fi
```

Can only be used before `\makeglossaries`

```
429 \onlypremakeg\glsSetAlphaCompositor
```

`\gls@suffixF` Suffix to use for a two page list. This overrides the separator and the closing page number if set to something other than an empty macro.

```
430 \newcommand*\gls@suffixF{-}
```

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

```
431 \newcommand*\glsSetSuffixF[1]{%
```

```
432 \renewcommand*\gls@suffixF{#1}}
```

Only has an effect when used before `\makeglossaries`

```
433 \onlypremakeg\glsSetSuffixF
```

`\gls@suffixFF` Suffix to use for a three page list. This overrides the separator and the closing page number if set to something other than an empty macro.

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

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

```
435 \newcommand*\glsSetSuffixFF[1]{%
```

```
436 \renewcommand*\gls@suffixFF{#1}%
```

```
437 }
```

`\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".

```
438 \ifcsundef{hyperlink}%
```

```
439 {%
```

```
440 \newcommand*\glsnumberformat[1]{#1}%
```

```
441 }%
```

```
442 {%
```

```
443 \newcommand*\glsnumberformat[1]{\glshypernumber{#1}}%
```

```
444 }
```

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`

```
445 \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`

```
446 \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`

```
447 \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`

```
448 \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`.

```
449 \newcommand*\glossarysection}[2][\@gls@title]{%
450   \def\@gls@title{#2}%
451   \ifcsundef{phantomsection}%
452     {%
453       \@glossarysection{#1}{#2}%
454     }%
455   {%
456     \p@glossarysection{#1}{#2}%
457   }%
458   \glossarymark{\glossarytoctitle}%
459 }
```

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

```
460 \ifcsundef{glossarymark}%
461 {%
462   \newcommand{\glossarymark}[1]{\@mkboth{#1}{#1}}
```

```

463 }%
464 {%
465   \GlossariesWarning{overriding \string\glossarymark}%
466   \@ifclassloaded{memoir}%
467   {
468     \renewcommand{\glossarymark}[1]{%
469       \markboth{\memUHead{#1}}{\memUHead{#1}}%
470     }
471   }
472   {
473     \renewcommand{\glossarymark}[1]{\@mkboth{#1}{#1}}
474   }
475 }

```

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`

```

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

```

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

`\@glossarysection`

```

478 \newcommand*\@glossarysection[2]{%
479 \ifx\@glossarysecstar\@empty
480   \csname\@glossarysec\endcsname{#2}%
481 \else
482   \csname\@glossarysec\endcsname*{#2}%
483   \@gls@toc{#1}{\@glossarysec}%
484 \fi
485 \@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`

```

486 \newcommand*\@pglossarysection[2]{%
487 \glsclearpage
488 \phantomsection
489 \ifx\@glossarysecstar\@empty
490   \csname\@glossarysec\endcsname{#2}%
491 \else
492   \@gls@toc{#1}{\@glossarysec}%
493   \csname\@glossarysec\endcsname*{#2}%

```

```
494 \fi
495 \@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.

```
496 \newcommand*\gls@doclearpage}{%
497   \ifthenelse\equal{\@glossarysec}{chapter}}{%
498   {%
499     \ifcsundef{cleardoublepage}{\clearpage}{\cleardoublepage}%
500   }%
501   {}}%
502 }
```

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

```
503 \newcommand*\glsclearpage}{\gls@doclearpage}
```

The glossary is added to the table of contents if `glstoc` flag set. If it is set, `\@gls@toc` will add a line to the `.toc` file, otherwise it will do nothing. (The first argument to `\@gls@toc` is the title for the table of contents, the second argument is the sectioning type.)

```
\@gls@toc
504 \newcommand*\@gls@toc[2]{%
505   \ifglstoc
506     \ifglsnumberline
507       \addcontentsline{toc}{#2}{\numberline{#1}}%
508     \else
509       \addcontentsline{toc}{#2}{#1}%
510   \fi
511 }
```

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

```
512 \newcommand*\glsnoxindywarning[1]{%
513   \GlossariesWarning{Not in xindy mode --- ignoring \string#1}%
514 }
```

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

```
515 \ifglsxindy
516   \edef\@xdyattributes{\string"default\string"}%
517 \fi
```

```

\@xdyattributelist  Comma-separated list of attributes.
518 \ifglxindy
519 \edef\@xdyattributelist{}%
520 \fi

\@xdylocref  Define list of markup location references.
521 \ifglxindy
522 \def\@xdylocref{}
523 \fi

\@gls@ifinlist
524 \newcommand*\@gls@ifinlist[4]{%
525 \def\@do@ifinlist##1,#1,##2\end@ifinlist{%
526 \def\@gls@listsuffix{##2}%
527 \ifx\@gls@listsuffix\@empty
528 #4%
529 \else
530 #3%
531 \fi
532 }%
533 \@do@ifinlist,#2,#1,\end@ifinlist
534 }

\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.
535 \ifglxindy
536 \newcommand*\@xdycounters{\@glscounter}
537 \newcommand*\GlsAddXdyCounters[1]{%
538 \@for\@gls@ctr:=#1\do{%
Check if already in list before adding.
539 \edef\@do@addcounter{%
540 \noexpand\@gls@ifinlist{\@gls@ctr}{\@xdycounters}{}%
541 {%
542 \noexpand\edef\noexpand\@xdycounters{\@xdycounters,%
543 \noexpand\@gls@ctr}%
544 }%
545 }%
546 \@do@addcounter
547 }
548 }

Only has an effect before \writeist:
549 \@onlypremakeg\GlsAddXdyCounters
550 \else
551 \newcommand*\GlsAddXdyCounters[1]{%
552 \glsnoxindywarning\GlsAddXdyAttribute
553 }
554 \fi

```

```

\@disabled@glSaddxdycounters Counters must all be identified before adding attributes.
555 \newcommand*\@disabled@glSaddxdycounters{%
556   \PackageError{glossaries}{\string\GlSAddXdyCounters\space
557   can't be used after \string\GlSAddXdyAttribute}{Move all
558   occurrences of \string\GlSAddXdyCounters\space before the first
559   instance of \string\GlSAddXdyAttribute}%
560 }

\GlSAddXdyAttribute Adds an attribute.
561 \ifglSxindy
  First define internal command that adds an attribute for a given counter (2nd
  argument is the counter):
562   \newcommand*\@glSaddxdyattribute[2]{%
  Add to xindy attribute list
563     \edef\@xdyattributes{\@xdyattributes ^^J \string"#1\string" ^^J
564     \string"#2#1\string"}%
  Add to xindy markup location.
565     \expandafter\toks@\expandafter{\@xdylocref}%
566     \edef\@xdylocref{\the\toks@ ^^J%
567     (markup-locref
568     :open \string"\string~n%
569     \expandafter\string\csname glSX#2X#1\endcsname
570     \string" ^^J
571     :close \string"\string" ^^J
572     :attr \string"#2#1\string")}%
  Define associated attribute command \glSX{counter}X{attribute}{\langle Hprefix\rangle}{\langle n\rangle}
573     \expandafter\gdef\csname glSX#2X#1\endcsname##1##2{%
574     \setentrycounter[##1]{##2}\csname #1\endcsname{##2}%
575     }%
576   }
  High-level command:
577   \newcommand*\GlSAddXdyAttribute[1]{%
  Add to comma-separated attribute list
578     \ifx\@xdyattributelist\empty
579     \edef\@xdyattributelist{#1}%
580     \else
581     \edef\@xdyattributelist{\@xdyattributelist,#1}%
582     \fi
  Iterate through all specified counters and add counter-dependent attributes:
583     \@for\@this@counter:=\@xdycounters\do{%
584     \protected\edef\glS@do@addxdyattribute{%
585     \noexpand\@glSaddxdyattribute{#1}{\@this@counter}%
586     }
587     \glS@do@addxdyattribute
588     }%

```

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

```
589 \let\GlsAddXdyCounters\@disabled@glssaddxdycounters
590 }
```

Only has an effect before `\writeist`:

```
591 \@onlypremakeg\GlsAddXdyAttribute
592 \else
593 \newcommand*\GlsAddXdyAttribute[1]{%
594 \glsnoinxdywarning\GlsAddXdyAttribute}
595 \fi
```

`\@gls@addpredefinedattributes` Add known attributes for all defined counters

```
596 \ifglsexindy
597 \newcommand*\@gls@addpredefinedattributes}{%
598 \GlsAddXdyAttribute{glsnumberformat}
599 \GlsAddXdyAttribute{textrm}
600 \GlsAddXdyAttribute{textsf}
601 \GlsAddXdyAttribute{texttt}
602 \GlsAddXdyAttribute{textbf}
603 \GlsAddXdyAttribute{textmd}
604 \GlsAddXdyAttribute{textit}
605 \GlsAddXdyAttribute{textup}
606 \GlsAddXdyAttribute{textsl}
607 \GlsAddXdyAttribute{textsc}
608 \GlsAddXdyAttribute{emph}
609 \GlsAddXdyAttribute{glshypernumber}
610 \GlsAddXdyAttribute{hyperrm}
611 \GlsAddXdyAttribute{hypersf}
612 \GlsAddXdyAttribute{hypertt}
613 \GlsAddXdyAttribute{hyperbf}
614 \GlsAddXdyAttribute{hypermd}
615 \GlsAddXdyAttribute{hyperit}
616 \GlsAddXdyAttribute{hyperup}
617 \GlsAddXdyAttribute{hypersl}
618 \GlsAddXdyAttribute{hypersc}
619 \GlsAddXdyAttribute{hyperemph}
620 }
621 \else
622 \let\@gls@addpredefinedattributes\relax
623 \fi
```

`\@xdyuseralphabets` List of additional alphabets

```
624 \def\@xdyuseralphabets{}
```

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

```
625 \ifglsexindy
626 \newcommand*\GlsAddXdyAlphabet}[2]{%
627 \edef\@xdyuseralphabets{%
```

```

628 \@xdyuseralphabets ^^J
629 (define-alphabet "#1" (#2))}
630 \else
631 \newcommand*{\GlsAddXdyAlphabet}[2]{%
632 \glsnoxindywarning\GlsAddXdyAlphabet}
633 \fi

```

This code is only required for xindy:

```
634 \ifglsxindy
```

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

```

635 \newcommand*{\@gls@xdy@locationlist}{%
636 roman-page-numbers,%
637 Roman-page-numbers,%
638 arabic-page-numbers,%
639 alpha-page-numbers,%
640 Alpha-page-numbers,%
641 Appendix-page-numbers,%
642 arabic-section-numbers%
643 }

```

Each location class *<name>* has the format stored in `\@gls@xdy@Lclass@<name>`.
Set up predefined formats.

`\@xdy@Lclass@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.

```

644 \protected@edef\@gls@roman{\@roman{0}\string"
645 \string"roman-numbers-lowercase\string" :sep \string"}}%
646 \@onelevel@sanitize\@gls@roman
647 \edef\@tmp{\string" \string"roman-numbers-lowercase\string"
648 :sep \string"}%
649 \@onelevel@sanitize\@tmp
650 \ifx\@tmp\@gls@roman
651 \expandafter
652 \edef\csname @gls@xdy@Lclass@roman-page-numbers\endcsname{%
653 \string"roman-numbers-lowercase\string"%
654 }%
655 \else
656 \expandafter
657 \edef\csname @gls@xdy@Lclass@roman-page-numbers\endcsname{
658 :sep \string"\@gls@roman\string"%
659 }%
660 \fi

```

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

```

661 \expandafter\def\csname @gls@xdy@Lclass@Roman-page-numbers\endcsname{%
662 \string"roman-numbers-uppercase\string"%
663 }%

```

```

\@xdy@Lclass@arabic-page-numbers Arabic numbers (1, 2, ...).
664 \expandafter\def\csname @gls@xdy@Lclass@arabic-page-numbers\endcsname{%
665 \string"arabic-numbers\string"%
666 }%

\@xdy@Lclass@alpha-page-numbers Lower case alphabetical (a, b, ...).
667 \expandafter\def\csname @gls@xdy@Lclass@alpha-page-numbers\endcsname{%
668 \string"alpha\string"%
669 }%

\@xdy@Lclass@Alpha-page-numbers Upper case alphabetical (A, B, ...).
670 \expandafter\def\csname @gls@xdy@Lclass@Alpha-page-numbers\endcsname{%
671 \string"ALPHA\string"%
672 }%

\@xdy@Lclass@Appendix-page-numbers Appendix style locations (e.g. A-1, A-2, ..., B-1, B-2, ...). The separator is given
by \@glsAlphacompositor.
673 \expandafter\def\csname @gls@xdy@Lclass@Appendix-page-numbers\endcsname{%
674 \string"ALPHA\string"
675 :sep \string"\@glsAlphacompositor\string"
676 \string"arabic-numbers\string"%
677 }

\@Lclass@arabic-section-numbers Section number style locations (e.g. 1.1, 1.2, ...). The compositor is given by
\glscompositor.
678 \expandafter\def\csname @gls@xdy@Lclass@arabic-section-numbers\endcsname{%
679 \string"arabic-numbers\string"
680 :sep \string"\glscompositor\string"
681 \string"arabic-numbers\string"%
682 }%

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

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

End of xindy-only block:
685 \fi

\GlsAddXdyLocation \GlsAddXdyLocation[prefix-loc]{name}{definition} Define a new location
called name. The definition must use xindy syntax. (Note that this doesn't
check to see if the location is already defined. That is left to xindy to complain
about.)
686 \ifglsxindy
687 \newcommand*{\GlsAddXdyLocation}[3][{}]{%
688 \def\@gls@tmp{#1}%
689 \ifx\@gls@tmp\@empty

```

```

690     \edef\@xdyuserlocationdefs{%
691         \@xdyuserlocationdefs ^^J%
692         (define-location-class \string"#2\string"^^J\space\space
693         \space(:sep \string"{}\glsoopenbrace\string" #3
694             :sep \string"\glsclosebrace\string"))
695     }%
696 \else
697     \edef\@xdyuserlocationdefs{%
698         \@xdyuserlocationdefs ^^J%
699         (define-location-class \string"#2\string"^^J\space\space
700         \space(:sep "\glsoopenbrace"
701             #1
702             :sep "\glsclosebrace\glsoopenbrace" #3
703             :sep "\glsclosebrace"))
704     }%
705 \fi
706 \edef\@xdyuserlocationnames{%
707     \@xdyuserlocationnames^^J\space\space\space
708     \string"#1\string"}%
709 }

```

Only has an effect before `\writeist`:

```

710 \@onlypremakeg\GlsAddXdyLocation
711 \else
712     \newcommand*\GlsAddXdyLocation}[2]{%
713         \glsnoinindywarning\GlsAddXdyLocation}
714 \fi

```

`\@xdylocationclassorder` Define location class order

```

715 \ifglxindy
716     \edef\@xdylocationclassorder{^^J\space\space\space
717         \string"roman-page-numbers\string"^^J\space\space\space
718         \string"arabic-page-numbers\string"^^J\space\space\space
719         \string"arabic-section-numbers\string"^^J\space\space\space
720         \string"alpha-page-numbers\string"^^J\space\space\space
721         \string"Roman-page-numbers\string"^^J\space\space\space
722         \string"Alpha-page-numbers\string"^^J\space\space\space
723         \string"Appendix-page-numbers\string"
724         \@xdyuserlocationnames^^J\space\space\space
725         \string"see\string"
726     }
727 \fi

```

Change the location order.

`\GlsSetXdyLocationClassOrder`

```

728 \ifglxindy
729     \newcommand*\GlsSetXdyLocationClassOrder[#1]{%
730         \def\@xdylocationclassorder{#1}}
731 \else

```

```

732 \newcommand*\GlsSetXdyLocationClassOrder[1]{%
733 \glsnoxywarning\GlsSetXdyLocationClassOrder}
734 \fi

\@xdysortrules Define sort rules
735 \ifglxindy
736 \def\@xdysortrules{}
737 \fi

\GlsAddSortRule Add a sort rule
738 \ifglxindy
739 \newcommand*\GlsAddSortRule[2]{%
740 \expandafter\toks@\expandafter{\@xdysortrules}%
741 \protected@edef\@xdysortrules{\the\toks@ ^^J
742 (sort-rule \string"#1\string" \string"#2\string")}%
743 }
744 \else
745 \newcommand*\GlsAddSortRule[2]{%
746 \glsnoxywarning\GlsAddSortRule}
747 \fi

\@xdyrequiredstyles Define list of required styles (this should be a comma-separated list of xindy
styles)
748 \ifglxindy
749 \def\@xdyrequiredstyles{tex}
750 \fi

\GlsAddXdyStyle Add a xindy style to the list of required styles
751 \ifglxindy
752 \newcommand*\GlsAddXdyStyle[1]{%
753 \edef\@xdyrequiredstyles{\@xdyrequiredstyles,#1}}%
754 \else
755 \newcommand*\GlsAddXdyStyle[1]{%
756 \glsnoxywarning\GlsAddXdyStyle}
757 \fi

\GlsSetXdyStyles Reset the list of required styles
758 \ifglxindy
759 \newcommand*\GlsSetXdyStyles[1]{%
760 \edef\@xdyrequiredstyles{#1}}
761 \else
762 \newcommand*\GlsSetXdyStyles[1]{%
763 \glsnoxywarning\GlsSetXdyStyles}
764 \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.
765 \ifglxindy
766 \@ifpackageloaded{babel}{%

```

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

```

767 \def\findrootlanguage{\begingroup
768   \escapechar=-1\relax
   normalize \languagename to category 12 chars
769   \edef\languagename{%
770     \expandafter\string\csname\languagename\endcsname}%
   disable babel.sty useless commands
771   \def\NeedsTeXFormat##1[##2]{}%
772   \def\ProvidesPackage##1[##2]{}%
773   \let\LdfInit\relax
774   \def\languageattribute##1##2{%
   change the meaning of \DeclareOption
775   \def\DeclareOption##1##2{%
   at \DeclareOption* we end
776     \ifx##1*\expandafter\endinput\else
   else we build a string with the first argument
777     \edef\testlanguage{\expandafter\string\csname##1\endcsname}%
   if \testlanguage and \languagename are the same we execute the second argu-
   ment
778     \ifx\testlanguage\languagename##2\fi
779     \fi}
   almost all options of babel are \input{<name>.ldf}
780 \def\input##1{\stripldf##1}%
   we put the root language name in \rootlanguagename
781 \def\stripldf##1.ldf{\gdef\rootlanguagename{##1}}%
   now input babel.sty, using the primitive \input
782 @@input babel.sty
783 \endgroup}%
784 }{%
   hasn't been loaded, so check if has been loaded
785 \ifpackageloaded{ngerman}{%
786   \def\findrootlanguage{%
787     \def\rootlanguagename{german}}%
788   }{%
   Neither babel nor ngerman have been loaded, so assume the root language is English
789   \def\findrootlanguage{%
790     \def\rootlanguagename{english}}%
791   }%
792 }%
793 \fi

```

`\rootlanguage` Set default root language to English.
794 `\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.
795 `\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.
796 `\ifglxindy`
797 `\newcommand*\GlsSetXdyLanguage[2][\glsdefaulttype]{%`
798 `\ifglossaryexists{#1}{%`
799 `\expandafter\def\csname @xdy@#1@language\endcsname{#2}%`
800 `}{%`
801 `\PackageError{glossaries}{Can't set language type for`
802 `glossary type '#1' --- no such glossary}{%`
803 `You have specified a glossary type that doesn't exist}}`
804 `\else`
805 `\newcommand*\GlsSetXdyLanguage[2] []{%`
806 `\glsnoxywarning\GlsSetXdyLanguage}`
807 `\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.
808 `\def\@gls@codepage#1#2{}`

`\GlsSetXdyCodePage` Define command to set the code page.
809 `\ifglxindy`
810 `\newcommand*\GlsSetXdyCodePage[1]{%`
811 `\renewcommand*\gls@codepage{#1}%`
812 `}`
813 `\else`
814 `\newcommand*\GlsSetXdyCodePage[1]{%`
815 `\glsnoxywarning\GlsSetXdyCodePage}`
816 `\fi`

`\@xdylettergroups` Store letter group definitions.
817 `\ifglxindy`
818 `\ifgls@xindy@glsnumbers`
819 `\def\@xdylettergroups{(define-letter-group`
820 `\string"glnumbers\string"^^J\space\space\space`
821 `:prefixes (\string"0\string" \string"1\string"`
822 `\string"2\string" \string"3\string" \string"4\string"`

```

823     \string"5\string" \string"6\string" \string"7\string"
824     \string"8\string" \string"9\string")^^J\space\space\space
825     :before \string"\@glsfirstletter\string"})}
826 \else
827   \def\@xdylettergroups{}
828 \fi
829 \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.

```

830 \newcommand*\GlsAddLetterGroup[2]{%
831   \expandafter\toks@\expandafter{\@xdylettergroups}%
832   \protected@edef\@xdylettergroups{\the\toks@^^J%
833   (define-letter-group \string"#1\string"^^J\space\space\space#2)}%
834 }%

```

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.

```

835 \newcommand*\forallglossaries[3][\@glo@types]{%
836   \@for#2:=#1\do{\ifx#2\@empty\else#3\fi}%
837 }

```

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

```
\forallglsentries[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.

```

838 \newcommand*\forallglsentries[3][\glsdefaulttype]{%
839   \edef\@glo@list{\csname glolist@#1\endcsname}%
840   \@for#2:=\@glo@list\do{\ifx#2\@empty\else#3\fi}%
841 }

```

`\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@`.

```

842 \newcommand*\forallglsentries[3][\@glo@types]{%
843   \expandafter\forallglossaries\expandafter[#1]{\@this@glo@}{%
844   \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.

```
845 \newcommand{\ifglossaryexists}[3]{%
846   \ifcsundef{@glo#1@out}{#3}{#2}%
847 }
```

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

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

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

```
848 \newcommand{\ifglentryexists}[3]{%
849   \ifcsundef{glo#1@name}{#3}{#2}%
850 }
```

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

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

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

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

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

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

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

```
852 \newcommand{\glstoifexists}[2]{%
853   \ifglentryexists{#1}{#2}{%
854     \PackageError{glossaries}{Glossary entry '#1' has not been
855     defined}{You need to define a glossary entry before you
856     can use it.}}%
857 }
```

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

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

```
858 \newcommand{\glstoifnoexists}[2]{%
859   \ifglentryexists{#1}{%
860     \PackageError{glossaries}{Glossary entry '#1' has already
861     been defined}{}}{#2}%
862 }
```

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`

```
863 \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`

```
864 \newcommand*{\newglossary}[5][glg]{%
865 \ifglossaryexists{#2}{%
866 \PackageError{glossaries}{Glossary type ‘#2’ already exists}{%
867 You can’t define a new glossary called ‘#2’ because it already
868 exists}%
869 }{%
```

Check if default has been set

```
870 \ifx\glsdefaulttype\relax
871 \gdef\glsdefaulttype{#2}%
872 \fi
```

Add this to the list of glossary types:

```
873 \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.

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

Store details of this new glossary type:

```
875 \expandafter\def\csname @glotype@#2@in\endcsname{#3}%
876 \expandafter\def\csname @glotype@#2@out\endcsname{#4}%
877 \expandafter\def\csname @glotype@#2@title\endcsname{#5}%
878 \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.

```

879 \ifcsundef{gls@#2@display}%
880 {%
881   \expandafter\gdef\csname gls@#2@display\endcsname{\glsdisplay}%
882 }%
883 {}%
884 \ifcsundef{gls@#2@displayfirst}%
885 {%
886   \expandafter\gdef\csname gls@#2@displayfirst\endcsname{%
887     \glsdisplayfirst
888   }%
889 }%
890 {}%
```

Define sort counter if required:

```

891 \@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.)
892 \@ifnextchar[{\@gls@setcounter{#2}}%
893   {\@gls@setcounter{#2}[\glscounter]}%
894 }
```

`\altnewglossary`

```

895 \newcommand*{\altnewglossary}[3]{%
896   \newglossary[#2-glg]{#1}{#2-gls}{#2-glo}{#3}%
897 }
```

Only define new glossaries in the preamble:

```

898 \@onlypreamble{\newglossary}
```

Only define new glossaries before `\makeglossaries`

```

899 \@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 L^AT_EX, `\@newglossary` simply ignores its arguments.

`\@newglossary`

```

900 \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`

```

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

Add counter to xindy list, if not already added:

```
903 \ifglxindy
904   \GlsAddXdyCounters{#2}%
905 \fi
906 }
```

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

`\@gls@getcounter`

```
907 \newcommand*{\@gls@getcounter}[1]{%
908 \csname @glo@type@#1@counter\endcsname}
```

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

```
909 \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.

```
910 \define@key{glossentry}{name}{%
911 \def\@glo@name{#1}%
912 }
```

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.)

```
913 \define@key{glossentry}{description}{%
914 \def\@glo@desc{#1}%
915 }
```

`descriptionplural`

```
916 \define@key{glossentry}{descriptionplural}{%
```

```

917 \def\@glo@descplural{#1}%
918 }

sort The sort key needs to be sanitized here (the sort key is provided for makeindex's
benefit, not for use in the document). The sort key is optional when defining a
new glossary entry. If omitted, the value is given by  $\langle name \rangle \langle description \rangle$ .
919 \define@key{glossentry}{sort}{%
920 \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.
921 \define@key{glossentry}{text}{%
922 \def\@glo@text{#1}%
923 }

plural The plural key determines how the plural form of the term should be dis-
played in the document. If omitted, the plural is constructed by appending
\glspluralsuffix to the value of the text key.
924 \define@key{glossentry}{plural}{%
925 \def\@glo@plural{#1}%
926 }

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.
927 \define@key{glossentry}{first}{%
928 \def\@glo@first{#1}%
929 }

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.
930 \define@key{glossentry}{firstplural}{%
931 \def\@glo@firstplural{#1}%
932 }

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 sym-
bol, 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).
933 \define@key{glossentry}{symbol}{%
934 \def\@glo@symbol{#1}%
935 }

```

symbolplural

```
936 \define@key{glossentry}{symbolplural}{%
937 \def\@glo@symbolplural{#1}%
938 }
```

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

```
939 \define@key{glossentry}{type}{%
940 \def\@glo@type{#1}}
```

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

```
941 \define@key{glossentry}{counter}{%
942 \ifcsundef{c@#1}%
943 {%
944 \PackageError{glossaries}%
945 {There is no counter called ‘#1’}%
946 {%
947 The counter key should have the name of a valid counter
948 as its value%
949 }%
950 }%
951 {%
952 \def\@glo@counter{#1}%
953 }%
954 }
```

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

```
955 \define@key{glossentry}{see}{%
956 \def\@glo@see{#1}%
957 \@glo@seeautonumberlist
958 }
```

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

```
959 \define@key{glossentry}{parent}{%
960 \def\@glo@parent{#1}}
```

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

```
961 \define@choicekey{glossentry}{nonumberlist}{\val\nr}{true,false}[true]{%
962 \ifcase\nr\relax
963 \def\@glo@prefix{\glsnonextpages}%
964 \else
965 \def\@glo@prefix{\glsnextpages}%
966 \fi
967 }
```

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

```

user1
968 \define@key{glossentry}{user1}{%
969 \def\@glo@useri{#1}%
970 }

user2
971 \define@key{glossentry}{user2}{%
972 \def\@glo@userii{#1}%
973 }

user3
974 \define@key{glossentry}{user3}{%
975 \def\@glo@useriii{#1}%
976 }

user4
977 \define@key{glossentry}{user4}{%
978 \def\@glo@useriv{#1}%
979 }

user5
980 \define@key{glossentry}{user5}{%
981 \def\@glo@userv{#1}%
982 }

user6
983 \define@key{glossentry}{user6}{%
984 \def\@glo@uservi{#1}%
985 }

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.
986 \define@key{glossentry}{short}{%
987 \def\@glo@short{#1}%
988 }

shortplural This key is provided for use by \newacronym.
989 \define@key{glossentry}{shortplural}{%
990 \def\@glo@shortpl{#1}%
991 }

long This key is provided for use by \newacronym.
992 \define@key{glossentry}{long}{%
993 \def\@glo@long{#1}%
994 }

longplural This key is provided for use by \newacronym.
995 \define@key{glossentry}{longplural}{%
996 \def\@glo@longpl{#1}%
997 }

```

```

\@glsnname Define command to generate error if name key is missing.
998 \newcommand*\@glsnname}{%
999 \PackageError{glossaries}{name key required in
1000 \string\newglossaryentry\space for entry ‘\@glo@label’}{You
1001 haven’t specified the entry name}}

\@glsdefaultplural Define command to set default plural.
1002 \newcommand*\@glsdefaultplural{\@glo@text\glspluralsuffix}

\@glsdefaultsort Define command to set default sort.
1003 \newcommand*\@glsdefaultsort{\@glo@name}

\gls@level Register to increment entry levels.
1004 \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.)
1005 \newrobustcmd\newglossaryentry}[2]{%
Check to see if this glossary entry has already been defined:
1006 \glsdoifnoexists{#1}{%
Store label
1007 \def\@glo@label{#1}%
Set up defaults. If the name or description keys are omitted, an error will be
generated.
1008 \let\@glo@name\@glsnname

1009 \def\@glo@desc{\PackageError{glossaries}{description key required in
1010 \string\newglossaryentry\space for entry ‘\@glo@label’}{You haven’t specified the entry descrip

1011 \def\@glo@descplural{\@glo@desc}%
1012 \def\@glo@type{\glsdefaulttype}%
1013 \def\@glo@symbol{\relax}%

1014 \def\@glo@symbolplural{\@glo@symbol}%
1015 \def\@glo@text{\@glo@name}%
1016 \let\@glo@plural\@glsdefaultplural

Using \let instead of \def to make later comparison avoid expansion issues.
(Thanks to Ulrich Diez for suggesting this.)
1017 \let\@glo@first\relax

1018 \let\@glo@firstplural\relax

Set the default sort:
1019 \let\@glo@sort\@glsdefaultsort

Set the default counter:
1020 \def\@glo@counter{\@gls@getcounter{\@glo@type}}%

```

```

1021 \def\@glo@see{}%
1022 \def\@glo@parent{}%
1023 \def\@glo@prefix{}%
1024 \def\@glo@useri{}%
1025 \def\@glo@userii{}%
1026 \def\@glo@useriii{}%
1027 \def\@glo@useriv{}%
1028 \def\@glo@userv{}%
1029 \def\@glo@uservi{}%
1030 \def\@glo@short{}%
1031 \def\@glo@shortpl{}%
1032 \def\@glo@long{}%
1033 \def\@glo@longpl{}%
    Add start hook in case another package wants to add extra keys.
1034 \@newglossaryentryprehook
    Extract key-val information from third parameter:
1035 \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.
1036 \ifcsundef{glolist@\@glo@type}%
1037 {%
1038   \PackageError{glossaries}%
1039   {Glossary type '\@glo@type' has not been defined}%
1040   {You need to define a new glossary type, before making entries
1041    in it}%
1042 }%
1043 {%
1044   \protected@edef\@glolist@{\csname glolist@\@glo@type\endcsname}%
1045   \expandafter\xdef\csname glolist@\@glo@type\endcsname{\@glolist@{#1},}%
1046 }%
    Initialise level to 0.
1047 \gls@level=0\relax
    Has this entry been assigned a parent?
1048 \ifx\@glo@parent\@empty
    Doesn't have a parent. Set \glo@<label>@parent to empty.
1049 \expandafter\gdef\csname glo@#1@parent\endcsname{}%
1050 \else
    Has a parent. Check to ensure this entry isn't its own parent.
1051 \ifthenelse{\equal{#1}{\@glo@parent}}{%
1052   \PackageError{glossaries}{Entry '#1' can't be its own parent}{}%
1053   \def\@glo@parent{}%
1054   \expandafter\gdef\csname glo@#1@parent\endcsname{}%
1055 }{%

```

```

Check the parent exists:
1056   \ifglentryexists{\@glo@parent}{%
Parent exists. Set \glo@<label>@parent.
1057   \expandafter\xdef\csname glo@#1@parent\endcsname{\@glo@parent}%
Determine level.
1058   \gls@level=\csname glo@\@glo@parent @level\endcsname\relax
1059   \advance\gls@level by 1\relax
If name hasn't been specified, use same as the parent name
1060   \ifx\@glo@name\@glsnoname
1061   \expandafter\let\expandafter\@glo@name
1062   \csname glo@\@glo@parent @name\endcsname
If name and plural haven't been specified, use same as the parent
1063   \ifx\@glo@plural\@glsdefaultplural
1064   \expandafter\let\expandafter\@glo@plural
1065   \csname glo@\@glo@parent @plural\endcsname
1066   \fi
1067   \fi
1068   }{%
Parent doesn't exist, so issue an error message and change this entry to have no
parent
1069   \PackageError{glossaries}{Invalid parent '@glo@parent'
1070   for entry '#1' - parent doesn't exist}{Parent entries
1071   must be defined before their children}%
1072   \def\@glo@parent{}%
1073   \expandafter\gdef\csname glo@#1@parent\endcsname{}%
1074   }%
1075   }%
1076 \fi
Set the level for this entry
1077 \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.)
1078 \ifx\relax\@glo@firstplural
1079   \ifx\relax\@glo@first
1080     \def\@glo@firstplural{\@glo@plural}%
1081     \def\@glo@first{\@glo@text}%
1082   \else
1083     \def\@glo@firstplural{\@glo@first\glspluralsuffix}%
1084   \fi
1085 \else
1086   \ifx\relax\@glo@first
1087     \def\@glo@first{\@glo@text}%

```

```

1088   \fi
1089 \fi
    Define commands associated with this entry:
1090 \expandafter
1091   \protected@xdef\csname glo@#1@text\endcsname{\@glo@text}%
1092 \expandafter
1093   \protected@xdef\csname glo@#1@plural\endcsname{\@glo@plural}%
1094 \expandafter
1095   \protected@xdef\csname glo@#1@first\endcsname{\@glo@first}%
1096 \expandafter
1097   \protected@xdef\csname glo@#1@firstpl\endcsname{\@glo@firstplural}%
1098 \expandafter
1099   \protected@xdef\csname glo@#1@type\endcsname{\@glo@type}%
1100 \expandafter
1101   \protected@xdef\csname glo@#1@counter\endcsname{\@glo@counter}%
1102 \expandafter
1103   \protected@xdef\csname glo@#1@useri\endcsname{\@glo@useri}%
1104 \expandafter
1105   \protected@xdef\csname glo@#1@userii\endcsname{\@glo@userii}%
1106 \expandafter
1107   \protected@xdef\csname glo@#1@useriii\endcsname{\@glo@useriii}%
1108 \expandafter
1109   \protected@xdef\csname glo@#1@useriv\endcsname{\@glo@useriv}%
1110 \expandafter
1111   \protected@xdef\csname glo@#1@userv\endcsname{\@glo@userv}%
1112 \expandafter
1113   \protected@xdef\csname glo@#1@uservi\endcsname{\@glo@uservi}%
1114 \expandafter
1115   \protected@xdef\csname glo@#1@short\endcsname{\@glo@short}%
1116 \expandafter
1117   \protected@xdef\csname glo@#1@shortpl\endcsname{\@glo@shortpl}%
1118 \expandafter
1119   \protected@xdef\csname glo@#1@long\endcsname{\@glo@long}%
1120 \expandafter
1121   \protected@xdef\csname glo@#1@longpl\endcsname{\@glo@longpl}%
1122 \@gls@sanitizename
1123 \expandafter\protected@xdef\csname glo@#1@name\endcsname{\@glo@name}%
    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.
1124 \def\@glo@@desc{\@glo@first}%
1125 \ifx\@glo@desc\@glo@@desc
1126   \let\@glo@desc\@glo@first
1127 \fi
1128 \@gls@sanitizedesc
1129 \expandafter\protected@xdef\csname glo@#1@desc\endcsname{\@glo@desc}%
1130 \expandafter\protected@xdef\csname glo@#1@descplural\endcsname{\@glo@descplural}%
    Set the sort key for this entry:
1131 \@gls@defsort{\@glo@type}{#1}%

```

```

1132 \def\@glo@@symbol{\@glo@text}%
1133 \ifx\@glo@symbol\@glo@@symbol
1134   \let\@glo@symbol\@glo@text
1135 \fi
1136 \@gls@sanitizesymbol
1137 \expandafter\protected@xdef\csname glo@#1@symbol\endcsname{\@glo@symbol}%
1138 \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):
1139 \expandafter\gdef\csname glo@#1@flagfalse\endcsname{%
1140 \expandafter\global\expandafter
1141 \let\csname ifglo@#1@flag\endcsname\iffalse}%
1142 \expandafter\gdef\csname glo@#1@flagtrue\endcsname{%
1143 \expandafter\global\expandafter
1144 \let\csname ifglo@#1@flag\endcsname\ifftrue}%
1145 \csname glo@#1@flagfalse\endcsname

Sort out any cross-referencing if required.
1146 \ifx\@glo@see\@empty
1147 \else
1148   \protected@edef\@do@glssee{%
1149     \noexpand\@gls@fixbraces\noexpand\@glo@list\@glo@see
1150     \noexpand\@nil
1151     \noexpand\expandafter\noexpand\@glssee\noexpand\@glo@list{#1}}%
1152   \do@glssee
1153 \fi
1154 }%

Determine and store main part of the entry's index format.
1155 \do@glo@storeentry{#1}%

Add end hook in case another package wants to add extra keys.
1156 \@newglossaryentryposthook
1157 }

\@newglossaryentryprehook Allow extra information to be added to glossary entries:
1158 \newcommand*{\@newglossaryentryprehook}{}

\@newglossaryentryposthook Allow extra information to be added to glossary entries:
1159 \newcommand*{\@newglossaryentryposthook}{}

\@glossaryentryfield Indicate what command should be used to display each entry in the glossary.
(This enables the glossaries-accsupp package to use \accsuppglossaryentryfield
instead.)
1160 \ifglsxindy
1161   \newcommand*{\@glossaryentryfield}{\string\@glossaryentryfield}
1162 \else
1163   \newcommand*{\@glossaryentryfield}{\string\glossaryentryfield}
1164 \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.)

```

1165 \ifglxindy
1166   \newcommand*{\@glossarysubentryfield}{%
1167     \string\@glossarysubentryfield}
1168 \else
1169   \newcommand*{\@glossarysubentryfield}{%
1170     \string\glossarysubentryfield}
1171 \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.)

```

1172 \newcommand{\@glo@storeentry}[1]{%
    Get the sort string and escape any special characters
1173 \protected@edef\@glo@sort{\csname glo@#1@sort\endcsname}%
1174 \@gls@checkmkidxchars\@glo@sort
    Same again for the name string.
1175 \protected@edef\@glo@name{\csname glo@#1@name\endcsname}%
1176 \@gls@checkmkidxchars\@glo@name
    Add the font command. (The backslash needs to be escaped for xindy.)
1177 \ifglxindy
1178   \protected@edef\@glo@name{\string\@glsnamefont{\@glo@name}}%
1179 \else
1180   \protected@edef\@glo@name{\string\glsnamefont{\@glo@name}}%
1181 \fi
    Get the description string and escape any special characters
1182 \protected@edef\@glo@desc{\csname glo@#1@desc\endcsname}%
1183 \@gls@checkmkidxchars\@glo@desc
    Same again for the symbol
1184 \protected@edef\@glo@symbol{\csname glo@#1@symbol\endcsname}%
1185 \@gls@checkmkidxchars\@glo@symbol
    Escape any special characters in the prefix
1186 \@gls@checkmkidxchars\@glo@prefix
    Get the parent, if one exists
1187 \edef\@glo@parent{\csname glo@#1@parent\endcsname}%
    Write the information to the glossary file.
1188 \ifglxindy
    Store using xindy syntax.
1189   \ifx\@glo@parent\@empty

```

```

Entry doesn't have a parent
1190   \expandafter\protected\xdef\csname glo@#1@index\endcsname{%
1191   (\string"\@glo@sort\string" %
1192   \string"\@glo@prefix\@glossaryentryfield{#1}{\@glo@name
1193   }{\@glo@desc}{\@glo@symbol}\string") %
1194   }%
1195   \else
Entry has a parent
1196   \expandafter\protected\xdef\csname glo@#1@index\endcsname{%
1197   \csname glo@\@glo@parent @index\endcsname
1198   (\string"\@glo@sort\string" %
1199   \string"\@glo@prefix\@glossarysubentryfield%
1200   {\csname glo@#1@level\endcsname}{#1}{\@glo@name
1201   }{\@glo@desc}{\@glo@symbol}\string") %
1202   }%
1203   \fi
1204   \else
Store using makeindex syntax.
1205   \ifx\@glo@parent\@empty
Sanitize \@glo@prefix
1206   \@onelevel@sanitize\@glo@prefix
Entry doesn't have a parent
1207   \expandafter\protected\xdef\csname glo@#1@index\endcsname{%
1208   \@glo@sort\@gls@actualchar\@glo@prefix
1209   \@glossaryentryfield{#1}{\@glo@name}{\@glo@desc
1210   }{\@glo@symbol}%
1211   }%
1212   \else
Entry has a parent
1213   \expandafter\protected\xdef\csname glo@#1@index\endcsname{%
1214   \csname glo@\@glo@parent @index\endcsname\@gls@levelchar
1215   \@glo@sort\@gls@actualchar\@glo@prefix
1216   \@glossarysubentryfield
1217   {\csname glo@#1@level\endcsname}{#1}{\@glo@name}{\@glo@desc
1218   }{\@glo@symbol}%
1219   }%
1220   \fi
1221 \fi
1222 }

```

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`

```
1223 \newcommand*\glsreset}[1]{%
1224 \glsdoifexists{#1}{%
1225 \expandafter\global\csname glo@#1@flagfalse\endcsname}}
```

As above, but with only a local effect:

`\glslocalreset`

```
1226 \newcommand*\glslocalreset}[1]{%
1227 \glsdoifexists{#1}{%
1228 \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`

```
1229 \newcommand*\glsunset}[1]{%
1230 \glsdoifexists{#1}{%
1231 \expandafter\global\csname glo@#1@flagtrue\endcsname}}
```

As above, but with only a local effect:

`\glslocalunset`

```
1232 \newcommand*\glslocalunset}[1]{%
1233 \glsdoifexists{#1}{%
1234 \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`

```
1235 \newcommand*\glsresetall}[1][\@glo@types]{%
1236 \forallglsentries[#1]{\@glsentry}{%
1237 \glsreset{\@glsentry}}}
```

As above, but with only a local effect:

`\glslocalresetall`

```
1238 \newcommand*\glslocalresetall}[1][\@glo@types]{%
1239 \forallglsentries[#1]{\@glsentry}{%
1240 \glslocalreset{\@glsentry}}}
```

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

`\glsunsetall`

```
1241 \newcommand*\glsunsetall}[1][\@glo@types]{%
1242 \forallglsentries[#1]{\@glsentry}{%
1243 \glsunset{\@glsentry}}}
```

As above, but with only a local effect:

```
\glslocalunsetall
1244 \newcommand*{\glslocalunsetall}[1][\@gls@types]{%
1245 \forallglsentries[#1]{\@glsentry}{%
1246 \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
1247 \newcommand*{\loadglsentries}[2][\@gls@default]{%
1248 \let\@gls@default\glsdefaulttype
1249 \def\glsdefaulttype{#1}\input{#2}%
1250 \let\glsdefaulttype\@gls@default}
```

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

```
1251 \@onlypreamble{\loadglsentries}
```

1.10 Using glossary entries in the text

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

```
\glsformat
1252 \newcommand*{\glsformat}[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

¹and any other valid L^AT_EX code that can be used in the preamble.

the final optional argument to commands like `\gls` and `\glspl`. The default is to display the first parameter followed by the additional text.

`\glsdisplayfirst`

```
1253 \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`

```
1254 \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`

```
1255 \newcommand*\defglsdisplay[2][\glsdefaulttype]{%
```

```
1256 \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`

```
1257 \newcommand*\defglsdisplayfirst[2][\glsdefaulttype]{%
```

```
1258 \expandafter\def\csname gls@#1@displayfirst\endcsname##1##2##3##4{#2}}
```

1.10.1 Links to glossary entries

The links to glossary entries all have a first optional argument that can be used to change the format and counter of the associated entry number. Except for `\glslink`, the commands like `\gls` have a final optional argument that can be used to insert additional text in the link (this will usually be appended, but can be redefined using `\defglsdisplay` and `\defglsdisplayfirst`). It goes against the L^AT_EX norm to have an optional argument after the mandatory arguments, but it makes more sense to write, say, `\gls[label]{}[s]` rather than,

say, `\gls[append='s]{label}`. Since these control sequences are defined to include the final square bracket, spaces will be ignored after them. This is likely to lead to confusion as most users would not expect, say, `\gls{<label>}` to ignore following spaces, so `\new@ifnextchar` from the package is required.

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

```

1259 \define@key{glslink}{counter}{%
1260   \ifcsundef{c@#1}%
1261   {%
1262     \PackageError{glossaries}%
1263     {There is no counter called '#1'}%
1264     {%
1265       The counter key should have the name of a valid counter
1266       as its value%
1267     }%
1268   }%
1269 }%
1270 \def\@gls@counter{#1}%
1271 }%
1272 }

```

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.

```

1273 \define@key{glslink}{format}{%
1274 \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.

```

1275 \define@boolkey{glslink}{hyper}[true]{}

```

Syntax:

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

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

There is also a starred version:

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

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

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

`\glslink`

```

1276 \newrobustcmd*{\glslink}{%
1277 \@ifstar\@sgls@link\@gls@link}

```

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

```
1278 \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.

```
1279 \newcommand*{\@gls@link}[3] []{%
```

```
1280   \ifglsentryexists{#2}%
```

```
1281   {%
```

```
1282     \@gls@link[#1]{#2}{#3}%
```

```
1283   }{%
```

```
1284     \PackageError{glossaries}{Glossary entry ‘#2’ has not been
```

```
1285     defined}{You need to define a glossary entry before you
```

```
1286     can use it.}%
```

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

```
1287     \glstextformat{#3}%
```

```
1288   }%
```

```
1289 }
```

`\@gls@link`

```
1290 \def\@gls@link[#1]#2#3{%
```

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

```
1291   \leavevmode
```

```
1292   \def\glslabel{#2}%
```

```
1293   \def\@glsnumberformat{glsnumberformat}%
```

```
1294   \edef\@gls@counter{\csname glo@#2@counter\endcsname}%
```

```
1295   \KV@glslink@hypertrue
```

```
1296   \setkeys{glslink}{#1}%
```

Store the entry's counter in `\theglsentrycounter`

```
1297   \@gls@saveentrycounter
```

Define sort key if necessary:

```
1298   \@gls@setsort{#2}%
```

```
1299   \@do@wrglossary{#2}%
```

```
1300   \ifKV@glslink@hyper
```

```
1301     \@glslink{glo:#2}{\glstextformat{#3}}%
```

```
1302   \else
```

```
1303     \glstextformat{#3}\relax
```

```
1304   \fi
```

```
1305 }
```

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

```
1306 \newcommand*{\@gls@saveentrycounter}{%
```

```
1307   \def\@gls@Hcounter{%
```

```

Are we using equation counter?
1308 \ifthenelse{\equal{\@gls@counter}{equation}}%
1309 {
    If we in align environment, \xatlevel@ will be defined. (Can't test for
    \@currenvir as may be inside an inner environment.)
1310 \ifcsundef{xatlevel@}%
1311 {%
1312 \edef\theglentrycounter{\expandafter\noexpand
1313 \csname the\@gls@counter\endcsname}%
1314 }%
1315 {%
1316 \ifx\xatlevel@\@empty
1317 \edef\theglentrycounter{\expandafter\noexpand
1318 \csname the\@gls@counter\endcsname}%
1319 \else
1320 \savecounters@
1321 \advance\c@equation by 1\relax
1322 \edef\theglentrycounter{\csname the\@gls@counter\endcsname}%
    Check if hyperref version of this counter
1323 \ifcsundef{theH\@gls@counter}%
1324 {%
1325 \def\@gls@Hcounter{\theglentrycounter}%
1326 }%
1327 {%
1328 \def\@gls@Hcounter{\csname theH\@gls@counter\endcsname}%
1329 }%
1330 \protected@edef\theHglentrycounter{\@gls@Hcounter}%
1331 \restorecounters@
1332 \fi
1333 }%
1334 }%
1335 {%
    Not using equation counter so no special measures:
1336 \edef\theglentrycounter{\expandafter\noexpand
1337 \csname the\@gls@counter\endcsname}%
1338 }%
    Check if hyperref version of this counter
1339 \ifx\@gls@Hcounter\@empty
1340 \ifcsundef{theH\@gls@counter}%
1341 {%
1342 \def\theHglentrycounter{\theglentrycounter}%
1343 }%
1344 {%
1345 \protected@edef\theHglentrycounter{\expandafter\noexpand
1346 \csname theH\@gls@counter\endcsname}%
1347 }%
1348 \fi

```

1349 }

`\@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.

```
1350 \def\@set@glo@numformat#1#2#3#4{%
1351   \expandafter\@glo@check@mkidxrangear#3\@nil
1352   \protected@edef#1{%
1353     \@glo@prefix setentrycounter[#4]{#2}%
1354     \expandafter\string\csname\@glo@suffix\endcsname
1355   }%
1356   \@gls@checkmkidxchars#1%
1357 }
```

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.

```
1358 \def\@glo@check@mkidxrangear#1#2\@nil{%
1359 \if#1(\relax
1360   \def\@glo@prefix{(%
1361   \if\relax#2\relax
1362     \def\@glo@suffix{glsnumberformat}%
1363   \else
1364     \def\@glo@suffix{#2}%
1365   \fi
1366 \else
1367 \if#1)\relax
1368   \def\@glo@prefix{)%
1369   \if\relax#2\relax
1370     \def\@glo@suffix{glsnumberformat}%
1371   \else
1372     \def\@glo@suffix{#2}%
1373   \fi
1374 \else
1375   \def\@glo@prefix{}\def\@glo@suffix{#1#2}%
1376 \fi
1377 \fi}
```

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

```
1378 \newcommand*\@gls@escbsdq[1]{%
1379   \def\@gls@checkedmkidx{%
1380     \let\gls@xdystring=#1\relax
1381     \@onelevel@sanitize\gls@xdystring
1382     \edef\do@gls@xdycheckbackslash{%
```

```

1383 \noexpand\@gls@xdycheckbackslash@gls@xdystring\noexpand\@nil
1384 \@backslashchar\@backslashchar\noexpand\null}%
1385 \do@gls@xdycheckbackslash
1386 \expandafter\@gls@updatechecked\@gls@checkedmkidx{\gls@xdystring}%
1387 \def\@gls@checkedmkidx{}%
1388 \expandafter\@gls@xdycheckquote@gls@xdystring\@nil""\null
1389 \expandafter\@gls@updatechecked\@gls@checkedmkidx{\gls@xdystring}%
1390 \let#1=\gls@xdystring
1391 }

```

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

\@gls@checkmkidxchars

```

1392 \newcommand{\@gls@checkmkidxchars}[1]{%
1393 \ifglsxindy
1394 \@gls@escbsdq{#1}%
1395 \else
1396 \def\@gls@checkedmkidx{}%
1397 \expandafter\@gls@checkquote#1\@nil""\null
1398 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
1399 \def\@gls@checkedmkidx{}%
1400 \expandafter\@gls@checkescquote#1\@nil""\null
1401 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
1402 \def\@gls@checkedmkidx{}%
1403 \expandafter\@gls@checkescactual#1\@nil\?\?\null
1404 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
1405 \def\@gls@checkedmkidx{}%
1406 \expandafter\@gls@checkactual#1\@nil??\null
1407 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
1408 \def\@gls@checkedmkidx{}%
1409 \expandafter\@gls@checkbar#1\@nil||\null
1410 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
1411 \def\@gls@checkedmkidx{}%
1412 \expandafter\@gls@checkeschar#1\@nil\\|\null
1413 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
1414 \def\@gls@checkedmkidx{}%
1415 \expandafter\@gls@checklevel#1\@nil!!\null
1416 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
1417 \fi
1418 }

```

Update the control sequence and strip trailing \@nil:

\@gls@updatechecked

```

1419 \def\@gls@updatechecked#1\@nil#2{\def#2{#1}}

```

\@gls@tmpb Define temporary token

```

1420 \newtoks\@gls@tmpb

```

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

```

1421 \def\@gls@checkquote#1"#2"#3\null{%
1422 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1423 \toks@={#1}%
1424 \ifx\null#2\null
1425 \ifx\null#3\null
1426 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1427 \def\@gls@checkquote{\relax}%
1428 \else
1429 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1430 \@gls@quotechar\@gls@quotechar\@gls@quotechar\@gls@quotechar}%
1431 \def\@gls@checkquote{\@gls@checkquote#3\null}%
1432 \fi
1433 \else
1434 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1435 \@gls@quotechar\@gls@quotechar}%
1436 \ifx\null#3\null
1437 \def\@gls@checkquote{\@gls@checkquote#2"\null}%
1438 \else
1439 \def\@gls@checkquote{\@gls@checkquote#2"#3\null}%
1440 \fi
1441 \fi
1442 \@gls@checkquote}

```

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

```

1443 \def\@gls@checkescquote#1"#2"#3\null{%
1444 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1445 \toks@={#1}%
1446 \ifx\null#2\null
1447 \ifx\null#3\null
1448 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1449 \def\@gls@checkescquote{\relax}%
1450 \else
1451 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1452 \@gls@quotechar\string"\@gls@quotechar
1453 \@gls@quotechar\string"\@gls@quotechar}%
1454 \def\@gls@checkescquote{\@gls@checkescquote#3\null}%
1455 \fi
1456 \else
1457 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1458 \@gls@quotechar\string"\@gls@quotechar}%
1459 \ifx\null#3\null
1460 \def\@gls@checkescquote{\@gls@checkescquote#2"\null}%
1461 \else
1462 \def\@gls@checkescquote{\@gls@checkescquote#2"#3\null}%
1463 \fi
1464 \fi
1465 \@gls@checkescquote}

```

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

```

1466 \def\@gls@checkescactual#1\?#2\?#3\null{%
1467 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1468 \toks@={#1}%
1469 \ifx\null#2\null
1470 \ifx\null#3\null
1471 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1472 \def\@gls@checkescactual{\relax}%
1473 \else
1474 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1475 \@gls@quotechar\string"\@gls@actualchar
1476 \@gls@quotechar\string"\@gls@actualchar}%
1477 \def\@gls@checkescactual{\@gls@checkescactual#3\null}%
1478 \fi
1479 \else
1480 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1481 \@gls@quotechar\string"\@gls@actualchar}%
1482 \ifx\null#3\null
1483 \def\@gls@checkescactual{\@gls@checkescactual#2\?\?\null}%
1484 \else
1485 \def\@gls@checkescactual{\@gls@checkescactual#2\?#3\null}%
1486 \fi
1487 \fi
1488 \@gls@checkescactual}

```

\@gls@checkescbar Similarly for \|:

```

1489 \def\@gls@checkescbar#1\|#2\|#3\null{%
1490 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1491 \toks@={#1}%
1492 \ifx\null#2\null
1493 \ifx\null#3\null
1494 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1495 \def\@gls@checkescbar{\relax}%
1496 \else
1497 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1498 \@gls@quotechar\string"\@gls@encapchar
1499 \@gls@quotechar\string"\@gls@encapchar}%
1500 \def\@gls@checkescbar{\@gls@checkescbar#3\null}%
1501 \fi
1502 \else
1503 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1504 \@gls@quotechar\string"\@gls@encapchar}%
1505 \ifx\null#3\null
1506 \def\@gls@checkescbar{\@gls@checkescbar#2\|\|\null}%
1507 \else
1508 \def\@gls@checkescbar{\@gls@checkescbar#2\|#3\null}%
1509 \fi
1510 \fi
1511 \@gls@checkescbar}

```

\@gls@checkesclevel Similarly for \!:

```

1512 \def\@gls@checkesclevel#1!#2!#3\null{%
1513 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1514 \toks@={#1}%
1515 \ifx\null#2\null
1516 \ifx\null#3\null
1517 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1518 \def\@gls@checkesclevel{\relax}%
1519 \else
1520 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1521 \@gls@quotechar\string"\@gls@levelchar
1522 \@gls@quotechar\string"\@gls@levelchar}%
1523 \def\@gls@checkesclevel{\@gls@checkesclevel#3\null}%
1524 \fi
1525 \else
1526 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1527 \@gls@quotechar\string"\@gls@levelchar}%
1528 \ifx\null#3\null
1529 \def\@gls@checkesclevel{\@gls@checkesclevel#2!\!\null}%
1530 \else
1531 \def\@gls@checkesclevel{\@gls@checkesclevel#2!#3\null}%
1532 \fi
1533 \fi
1534 \@gls@checkesclevel}

```

\@gls@checkbar and for |:

```

1535 \def\@gls@checkbar#1|#2|#3\null{%
1536 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1537 \toks@={#1}%
1538 \ifx\null#2\null
1539 \ifx\null#3\null
1540 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1541 \def\@gls@checkbar{\relax}%
1542 \else
1543 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1544 \@gls@quotechar\@gls@encapchar\@gls@quotechar\@gls@encapchar}%
1545 \def\@gls@checkbar{\@gls@checkbar#3\null}%
1546 \fi
1547 \else
1548 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1549 \@gls@quotechar\@gls@encapchar}%
1550 \ifx\null#3\null
1551 \def\@gls@checkbar{\@gls@checkbar#2||\null}%
1552 \else
1553 \def\@gls@checkbar{\@gls@checkbar#2|#3\null}%
1554 \fi
1555 \fi
1556 \@gls@checkbar}

```

\@gls@checklevel and for !:

```

1557 \def\@gls@checklevel#1!#2!#3\null{%
1558 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1559 \toks@={#1}%
1560 \ifx\null#2\null
1561 \ifx\null#3\null
1562 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1563 \def\@gls@checklevel{\relax}%
1564 \else
1565 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1566 \@gls@quotechar\@gls@levelchar\@gls@quotechar\@gls@levelchar}%
1567 \def\@gls@checklevel{\@gls@checklevel#3\null}%
1568 \fi
1569 \else
1570 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1571 \@gls@quotechar\@gls@levelchar}%
1572 \ifx\null#3\null
1573 \def\@gls@checklevel{\@gls@checklevel#2!!\null}%
1574 \else
1575 \def\@gls@checklevel{\@gls@checklevel#2!#3\null}%
1576 \fi
1577 \fi
1578 \@gls@checklevel}

```

\@gls@checkactual and for ?:

```

1579 \def\@gls@checkactual#1?#2?#3\null{%
1580 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1581 \toks@={#1}%
1582 \ifx\null#2\null
1583 \ifx\null#3\null
1584 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1585 \def\@gls@checkactual{\relax}%
1586 \else
1587 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1588 \@gls@quotechar\@gls@actualchar\@gls@quotechar\@gls@actualchar}%
1589 \def\@gls@checkactual{\@gls@checkactual#3\null}%
1590 \fi
1591 \else
1592 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1593 \@gls@quotechar\@gls@actualchar}%
1594 \ifx\null#3\null
1595 \def\@gls@checkactual{\@gls@checkactual#2??\null}%
1596 \else
1597 \def\@gls@checkactual{\@gls@checkactual#2?#3\null}%
1598 \fi
1599 \fi
1600 \@gls@checkactual}

```

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

```

1601 \def\@gls@xdycheckquote#1"#2"#3\null{%
1602 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%

```

```

1603 \toks@={#1}%
1604 \ifx\null#2\null
1605 \ifx\null#3\null
1606 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1607 \def\@gls@xdycheckquote{\relax}%
1608 \else
1609 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1610 \string"\string"}%
1611 \def\@gls@xdycheckquote{\@gls@xdycheckquote#3\null}%
1612 \fi
1613 \else
1614 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1615 \string"}%
1616 \ifx\null#3\null
1617 \def\@gls@xdycheckquote{\@gls@xdycheckquote#2"\null}%
1618 \else
1619 \def\@gls@xdycheckquote{\@gls@xdycheckquote#2"#3\null}%
1620 \fi
1621 \fi
1622 \@gls@xdycheckquote
1623 }

```

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

```

1624 \edef\def@gls@xdycheckbackslash{%
1625 \noexpand\def\noexpand\@gls@xdycheckbackslash##1\@backslashchar
1626 ##2\@backslashchar##3\noexpand\null{%
1627 \noexpand\@gls@tmpb=\noexpand\expandafter
1628 {\noexpand\@gls@checkedmkidx}%
1629 \noexpand\toks@={##1}%
1630 \noexpand\ifx\noexpand\null##2\noexpand\null
1631 \noexpand\ifx\noexpand\null##3\noexpand\null
1632 \noexpand\edef\noexpand\@gls@checkedmkidx{%
1633 \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@}%
1634 \noexpand\def\noexpand\@gls@xdycheckbackslash{\relax}%
1635 \noexpand\else
1636 \noexpand\edef\noexpand\@gls@checkedmkidx{%
1637 \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
1638 \@backslashchar\@backslashchar\@backslashchar\@backslashchar}%
1639 \noexpand\def\noexpand\@gls@xdycheckbackslash{%
1640 \noexpand\@gls@xdycheckbackslash##3\noexpand\null}%
1641 \noexpand\fi
1642 \noexpand\else
1643 \noexpand\edef\noexpand\@gls@checkedmkidx{%
1644 \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
1645 \@backslashchar\@backslashchar}%
1646 \noexpand\ifx\noexpand\null##3\noexpand\null
1647 \noexpand\def\noexpand\@gls@xdycheckbackslash{%
1648 \noexpand\@gls@xdycheckbackslash##2\@backslashchar
1649 \@backslashchar\noexpand\null}%

```

```

1650 \noexpand\else
1651 \noexpand\def\noexpand\@gls@xdycheckbackslash{%
1652 \noexpand\@gls@xdycheckbackslash##2\@backslashchar
1653 ##3\noexpand\null}%
1654 \noexpand\fi
1655 \noexpand\fi
1656 \noexpand\@gls@xdycheckbackslash
1657 }%
1658 }

```

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

```

1659 \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`.

```

1660 \ifcsundef{hyperlink}%
1661 {%
1662 \gdef@glslink#1#2{#2}%
1663 }%
1664 {%
1665 \gdef@glslink#1#2{\hyperlink{#1}{#2}}%
1666 }

```

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

```

1667 \newlength@gls@tmplen
1668 \ifcsundef{hypertarget}%
1669 {%
1670 \gdef@glstarget#1#2{#2}%
1671 }%
1672 {%
1673 \gdef@glstarget#1#2{%
1674 \settoheight@gls@tmplen{#2}%
1675 \raisebox@gls@tmplen{\hypertarget{#1}}{#2}%
1676 }%
1677 }

```

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

`\glsdisablehyper`

```

1678 \newcommand@glsdisablehyper{%
1679 \renewcommand*\@glslink[2]{##2}%
1680 \renewcommand*\@glstarget[2]{##2}}

```

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

`\glsenablehyper`

```
1681 \newcommand{\glsenablehyper}{%
1682 \renewcommand*\@glslink[2]{\hyperlink{##1}{##2}}%
1683 \renewcommand*\@glstarget[2]{%
1684 \settoheight{\gls@tmplen}{##2}%
1685 \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
```

```
1686 \newrobustcmd*{\gls}{\@ifstar\@sgls\@gls}
```

Define the starred form:

```
\@sgls
```

```
1687 \newcommand*{\@sgls}[1][\@gls[hyper=false,#1]]
```

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

```
\@gls
```

```
1688 \newcommand*{\@gls}[2][\@%
1689 \new@ifnextchar[{\@gls@{##1}{##2}}{\@gls@{##1}{##2}}]
```

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

```
1690 \def\@gls@#1#2[#3]{%
1691 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
Save options in \@gls@link@opts and label in \@gls@link@label
1692 \def\@gls@link@opts{#1}%
1693 \def\@gls@link@label{#2}%
```

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

```
1694 \ifglsused{#2}%
1695 {%
1696 \def\@glo@text{%
1697 \csname gls@\@glo@type @display\endcsname
```

```

1698     {\glsentrytext{#2}}{\glsentrydesc{#2}}{\glsentrysymbol{#2}}{#3}}%
1699 }%
1700 {%
1701   \def\@glo@text{%
1702     \csname gls@\@glo@type @displayfirst\endcsname
1703     {\glsentryfirst{#2}}{\glsentrydesc{#2}}{\glsentrysymbol{#2}}{#3}}%
1704 }%

```

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.

```

1705 \ifglsused{#2}{%
1706   \@gls@link[#1]{#2}{\@glo@text}%
1707 }{%
1708   \gls@checkisacronymlist\@glo@type
1709   \ifthenelse{(\boolean{glsisacronymlist}\AND
1710     \boolean{glsacrfootnote}) \OR \NOT\boolean{glshyperfirst}}{%
1711     \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
1712   }{%
1713     \@gls@link[#1]{#2}{\@glo@text}%
1714   }%
1715 }%

```

Indicate that this entry has now been used

```

1716 \glsunset{#2}}%
1717 }

```

`\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`

```

1718 \newrobustcmd*{\Gls}{\@ifstar\@sGls\@Gls}

```

Define the starred form:

```

1719 \newcommand*{\@sGls}[1][]{\@Gls[hyper=false,#1]}

```

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

```

1720 \newcommand*{\@Gls}[2][]{%

```

```

1721 \new@ifnextchar[{\@Gls@{#1}{#2}}{\@Gls@{#1}{#2} []]}

```

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

```

1722 \def\@Gls@#1#2[#3]{%

```

```

1723 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

1724 \def\@gls@link@opts{#1}%

```

```

1725 \def\@gls@link@label{#2}%

```

```

1726 \def\glslabel{#2}%

```

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

```
1727 \ifglsused{#2}%
1728 {%
1729   \protected@edef\@glo@text{%
1730     \csname gls@\@glo@type @display\endcsname
1731       {\glsentrytext{#2}}{\glsentrydesc{#2}}%
1732       {\glsentrysymbol{#2}}{#3}}%
1733 }%
1734 {%
1735   \protected@edef\@glo@text{%
1736     \csname gls@\@glo@type @displayfirst\endcsname
1737       {\glsentryfirst{#2}}{\glsentrydesc{#2}}%
1738       {\glsentrysymbol{#2}}{#3}}%
1739 }%
```

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.

```
1740 \ifglsused{#2}{%
1741   \@gls@link[#1]{#2}{%
1742     \expandafter\makefirstuc\expandafter{\@glo@text}}%
1743 }%
1744 \gls@checkisacronymlist\@glo@type
1745 \ifthenelse{(\boolean{@glsisacronymlist}\AND
1746   \boolean{glsacrfootnote}) \OR \NOT\boolean{glshyperfirst}}{%
1747   \@gls@link[#1,hyper=false]{#2}{%
1748     \expandafter\makefirstuc\expandafter{\@glo@text}}%
1749 }{%
1750   \@gls@link[#1]{#2}{%
1751     \expandafter\makefirstuc\expandafter{\@glo@text}}%
1752 }%
1753 }%
```

Indicate that this entry has now been used

```
1754 \glsunset{#2}}%
1755 }
```

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

\GLS

```
1756 \newrobustcmd*{\GLS}{\@ifstar\@sGLS\@GLS}
```

Define the starred form:

```
1757 \newcommand*{\@sGLS}[1] [] {\@GLS[hyper=false,#1]}
```

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

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

```
1759 \new@ifnextchar[{\@GLS@{#1}{#2}}{\@GLS@{#1}{#2} []}]}
```

```

\@GLS@ Read in the final optional argument:
1760 \def\@GLS@#1#2[#3]{%
1761 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
    Save options in \@gls@link@opts and label in \@gls@link@label
1762 \def\@gls@link@opts{#1}%
1763 \def\@gls@link@label{#2}%
    Determine what the link text should be (this is stored in \@glo@text).
1764 \ifglsused{#2}{\def\@glo@text{%
1765 \csname gls@\@glo@type @display\endcsname
1766 {\glsentrytext{#2}}{\glsentrydesc{#2}}{\glsentrysymbol{#2}}{#3}}}%
1767 \def\@glo@text{%
1768 \csname gls@\@glo@type @displayfirst\endcsname
1769 {\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.
1770 \ifglsused{#2}{%
1771   \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
1772 }{%
1773   \gls@checkisacronymlist\@glo@type
1774   \ifthenelse{(\boolean{\@glsisacronymlist})\AND
1775     \boolean{glsacrfootnote}) \OR \NOT\boolean{glshyperfirst}}{%
1776     \@gls@link[#1,hyper=false]{#2}{\MakeUppercase{\@glo@text}}%
1777   }{%
1778     \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
1779   }%
1780 }%
    Indicate that this entry has now been used
1781 \glsunset{#2}}%
1782 }

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

\glspl
1783 \newrobustcmd*{\glspl}{\@ifstar\@sglspl\@glspl}
    Define the starred form:
1784 \newcommand*{\@sglspl}[1][\@glspl[hyper=false,#1]]
    Defined the un-starred form. Need to determine if there is a final optional argu-
    ment
1785 \newcommand*{\@glspl}[2][\@glspl]{}
1786 \new@ifnextchar[{\@glspl@{#1}{#2}}{\@glspl@{#1}{#2}}{}

\@glspl@ Read in the final optional argument:
1787 \def\@glspl@#1#2[#3]{%
1788 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

1789 \def\@gls@link@opts{#1}%
1790 \def\@gls@link@label{#2}%

Determine what the link text should be (this is stored in \@glo@text)
1791 \ifglsused{#2}%
1792 {%
1793   \def\@glo@text{%
1794     \csname gls@\@glo@type @display\endcsname
1795       {\glsentryplural{#2}}{\glsentrydescplural{#2}}%
1796       {\glsentrysymbolplural{#2}}{#3}}%
1797 }%
1798 {%
1799   \def\@glo@text{%
1800     \csname gls@\@glo@type @displayfirst\endcsname
1801       {\glsentryfirstplural{#2}}{\glsentrydescplural{#2}}%
1802       {\glsentrysymbolplural{#2}}{#3}}%
1803 }%

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.
1804 \ifglsused{#2}{%
1805   \@gls@link[#1]{#2}{\@glo@text}%
1806 }{%
1807   \gls@checkisacronymlist\@glo@type
1808   \ifthenelse{(\boolean{@glsisacronymlist})\AND
1809     \boolean{glsacrfootnote}) \OR \NOT\boolean{glshyperfirst}}{%
1810     \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
1811   }{%
1812     \@gls@link[#1]{#2}{\@glo@text}%
1813   }%
1814 }%

Indicate that this entry has now been used
1815 \glsunset{#2}}%
1816 }

```

`\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`

```

1817 \newrobustcmd*{\Glspl}{\@ifstar\@sGlspl\@Glspl}

Define the starred form:
1818 \newcommand*{\@sGlspl}[1] [] {\@Glspl[hyper=false,#1]}

Defined the un-starred form. Need to determine if there is a final optional argu-
ment
1819 \newcommand*{\@Glspl}[2] [] {%
1820 \new@ifnextchar[{\@Glspl@{#1}{#2}}{\@Glspl@{#1}{#2} []}]

```

```

\@GLSp1@ Read in the final optional argument:
1821 \def\@GLSp1@#1#2[#3]{%
1822 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
    Save options in \@gls@link@opts and label in \@gls@link@label
1823 \def\@gls@link@opts{#1}%
1824 \def\@gls@link@label{#2}%
1825 \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.
1826 \ifglsused{#2}%
1827 {%
1828   \protected@edef\@glo@text{%
1829     \csname gls@\@glo@type @display\endcsname
1830     {\glsentryplural{#2}}{\glsentrydescplural{#2}}%
1831     {\glsentrysymbolplural{#2}}{#3}}%
1832 }%
1833 {%
1834   \protected@edef\@glo@text{%
1835     \csname gls@\@glo@type @displayfirst\endcsname
1836     {\glsentryfirstplural{#2}}{\glsentrydescplural{#2}}%
1837     {\glsentrysymbolplural{#2}}{#3}}%
1838 }%
    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.
1839 \ifglsused{#2}{%
1840   \@gls@link[#1]{#2}{%
1841     \expandafter\makefirstuc\expandafter{\@glo@text}}%
1842 }%
1843 \gls@checkisacronymlist\@glo@type
1844 \ifthenelse{(\boolean{@glsisacronymlist})\AND
1845   \boolean{glsacrfootnote}) \OR \NOT\boolean{gls hyperfirst}}{%
1846   \@gls@link[#1,hyper=false]{#2}{%
1847     \expandafter\makefirstuc\expandafter{\@glo@text}}%
1848 }{%
1849   \@gls@link[#1]{#2}{%
1850     \expandafter\makefirstuc\expandafter{\@glo@text}}%
1851 }%
1852 }%
    Indicate that this entry has now been used
1853 \glsunset{#2}}%
1854 }

\GLSp1 behaves like \glspl except that all the link text is converted to up-
percase.

\GLSp1
1855 \newrobustcmd*{\GLSp1}{\ifstar\@sGLSp1\@GLSp1}

```

Define the starred form:

```
1856 \newcommand*{\sGLSp1}[1] [] {\@GLSp1[hyper=false,#1]}
```

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

```
1857 \newcommand*{\@GLSp1}[2] [] {%
```

```
1858 \new@ifnextchar [ {\@GLSp1@{#1}{#2}} {\@GLSp1@{#1}{#2} []}]
```

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

```
1859 \def\@GLSp1@#1#2[#3] {%
```

```
1860 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%
```

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

```
1861 \def\@gls@link@opts{#1}%
```

```
1862 \def\@gls@link@label{#2}%
```

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

```
1863 \ifglsused{#2}{\def\@glo@text{%
```

```
1864 \csname gls@\@glo@type @display\endcsname
```

```
1865 {\glsentryplural{#2}}{\glsentrydescplural{#2}}{%
```

```
1866 \glsentrysymbolplural{#2}}{#3}}{%
```

```
1867 \def\@glo@text{%
```

```
1868 \csname gls@\@glo@type @displayfirst\endcsname
```

```
1869 {\glsentryfirstplural{#2}}{\glsentrydescplural{#2}}{%
```

```
1870 \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.

```
1871 \ifglsused{#2}{%
```

```
1872 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
```

```
1873 }{%
```

```
1874 \gls@checkisacronymlist\@glo@type
```

```
1875 \ifthenelse{(\boolean{\glsisacronymlist})\AND
```

```
1876 \boolean{glsacrfootnote}) \OR \NOT\boolean{glsyperfirst}}{%
```

```
1877 \@gls@link[#1,hyper=false]{#2}{\MakeUppercase{\@glo@text}}%
```

```
1878 }{%
```

```
1879 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
```

```
1880 }%
```

```
1881 }%
```

Indicate that this entry has now been used

```
1882 \glsunset{#2}}%
```

```
1883 }
```

`\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:

```
1884 \newrobustcmd*{\glsdisp}{\@ifstar\@sglsdisp\@glsdisp}
```

Define the starred form:

```
\@sgls
1885 \newcommand*{\@sglsdisp}[1] [] {\@sglsdisp[hyper=false,#1]}
```

Defined the un-starred form.

```
\@glsdisp
1886 \newcommand*{\@glsdisp}[3] [] {%
1887   \glsdoifexists{#2}{%
1888     \edef\@glo@type{\glsentrytype{#2}}%
1889     \def\@gls@link@opts{#1}%
1890     \def\@gls@link@label{#2}%
1891     \ifglsused{#2}%
1892     {%
1893       \def\@glo@text{%
1894         \csname gls@\@glo@type @display\endcsname
1895         {#3}{\glsentrydesc{#2}}{\glsentrysymbol{#2}}{}}%
1896       }%
1897     {%
1898       \def\@glo@text{%
1899         \csname gls@\@glo@type @displayfirst\endcsname
1900         {#3}{\glsentrydesc{#2}}{\glsentrysymbol{#2}}{}}%
1901       }%
1902       \ifglsused{#2}%
1903       {%
1904         \@gls@link[#1]{#2}{\@glo@text}%
1905       }%
1906       {%
1907         \gls@checkisacronymlist\@glo@type
1908         \ifthenelse{(\boolean{glsisacronymlist})\AND
1909           \boolean{glsacrfootnote}) \OR \NOT\boolean{gls hyperfirst}}%
1910         {%
1911           \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
1912         }%
1913         {%
1914           \@gls@link[#1]{#2}{\@glo@text}%
1915         }%
1916       }%
1917       \glsunset{#2}%
1918     }%
1919 }
```

Indicate that this entry has now been used

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

`\glstext`

```
1920 \newrobustcmd*{\glstext}{\@ifstar\@sGLstext\@GLstext}
```

Define the starred form:

```
1921 \newcommand*{\@sGLstext}[1] [] {\@GLstext[hyper=false,#1]}
```

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

```
1922 \newcommand*{\@GLstext}[2] [] {%
```

```
1923 \new@ifnextchar[{\@GLstext@{#1}{#2}}{\@GLstext@{#1}{#2}[]}]
```

Read in the final optional argument:

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

```
1925 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
1926 \protected@edef\@glo@text{\glsentrytext{#2}}%
```

Call `\@gls@link`

```
1927 \@gls@link[#1]{#2}{\@glo@text#3}%
```

```
1928 }%
```

```
1929 }
```

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

`\GLStext`

```
1930 \newrobustcmd*{\GLStext}{\@ifstar\@sGLStext\@GLStext}
```

Define the starred form:

```
1931 \newcommand*{\@sGLStext}[1] [] {\@GLStext[hyper=false,#1]}
```

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

```
1932 \newcommand*{\@GLStext}[2] [] {%
```

```
1933 \new@ifnextchar[{\@GLStext@{#1}{#2}}{\@GLStext@{#1}{#2}[]}]
```

Read in the final optional argument:

```
1934 \def\@GLStext@#1#2[#3] {%
```

```
1935 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
1936 \protected@edef\@glo@text{\glsentrytext{#2}}%
```

Call `\@gls@link`

```
1937 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
```

```
1938 }%
```

```
1939 }
```

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

`\Glstext`

```
1940 \newrobustcmd*{\Glstext}{\@ifstar\@sGlstext\@Glstext}
```

Define the starred form:

```
1941 \newcommand*{\@sGlstext}[1] [] {\@Glstext[hyper=false,#1]}
```

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

```
1942 \newcommand*{\@Glstext}[2] [] {%
```

```
1943 \new@ifnextchar[{\@Glstext@{#1}{#2}}{\@Glstext@{#1}{#2} []}]
```

Read in the final optional argument:

```
1944 \def\@Glstext@#1#2[#3] {%
```

```
1945 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%
```

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

```
1946 \protected@edef\@glo@text{\glsentrytext{#2}}%
```

Call `\@gls@link`

```
1947 \@gls@link[#1]{#2}{%
```

```
1948 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
```

```
1949 }%
```

```
1950 }
```

`\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`

```
1951 \newrobustcmd*{\glsfirst}{\@ifstar\@sglsfirst\@glsfirst}
```

Define the starred form:

```
1952 \newcommand*{\@sglsfirst}[1] [] {\@glsfirst[hyper=false,#1]}
```

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

```
1953 \newcommand*{\@glsfirst}[2] [] {%
```

```
1954 \new@ifnextchar[{\@glsfirst@{#1}{#2}}{\@glsfirst@{#1}{#2} []}]
```

Read in the final optional argument:

```
1955 \def\@glsfirst@#1#2[#3] {%
```

```
1956 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%
```

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

```
1957 \protected@edef\@glo@text{\glsentryfirst{#2}}%
```

Call `\@gls@link`

```
1958 \@gls@link[#1]{#2}{\@glo@text}#3}%
```

```
1959 }%
```

```
1960 }
```

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

`\Glsfirst`

```
1961 \newrobustcmd*{\Glsfirst}{\@ifstar\@sGlsfirst\@Glsfirst}

Define the starred form:
1962 \newcommand*{\@sGlsfirst}[1] [] {\@Glsfirst[hyper=false,#1]}
Defined the un-starred form. Need to determine if there is a final optional argu-
ment
1963 \newcommand*{\@Glsfirst}[2] [] {%
1964 \new@ifnextchar[{\@Glsfirst@{#1}{#2}}{\@Glsfirst@{#1}{#2} []}]

Read in the final optional argument:
1965 \def\@Glsfirst@#1#2[#3] {%
1966 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%
Determine what the link text should be (this is stored in \@glo@text)
1967 \protected@edef\@glo@text{\glsentryfirst{#2}}%
Call \@gls@link
1968 \@gls@link[#1]{#2}{%
1969 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
1970 }%
1971 }
```

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

`\GLSfirst`

```
1972 \newrobustcmd*{\GLSfirst}{\@ifstar\@sGLSfirst\@GLSfirst}

Define the starred form:
1973 \newcommand*{\@sGLSfirst}[1] [] {\@GLSfirst[hyper=false,#1]}
Defined the un-starred form. Need to determine if there is a final optional argu-
ment
1974 \newcommand*{\@GLSfirst}[2] [] {%
1975 \new@ifnextchar[{\@GLSfirst@{#1}{#2}}{\@GLSfirst@{#1}{#2} []}]

Read in the final optional argument:
1976 \def\@GLSfirst@#1#2[#3] {%
1977 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%
Determine what the link text should be (this is stored in \@glo@text)
1978 \protected@edef\@glo@text{\glsentryfirst{#2}}%
Call \@gls@link
1979 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}#3}%
1980 }%
1981 }
```

`\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`

```
1982 \newrobustcmd*{\glsplural}{\@ifstar\@sglsplural\@glsplural}
```

Define the starred form:

```
1983 \newcommand*{\sglsplural}[1] []{\@glsplural[hyper=false,#1]}
```

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

```
1984 \newcommand*{\@glsplural}[2] []{%
```

```
1985 \new@ifnextchar[{\@glsplural@{#1}{#2}}{\@glsplural@{#1}{#2} []}]
```

Read in the final optional argument:

```
1986 \def\@glsplural@#1#2[#3]{%
```

```
1987 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
1988 \protected@edef\@glo@text{\glsentryplural{#2}}%
```

Call \@gls@link

```
1989 \@gls@link[#1]{#2}{\@glo@text#3}%
```

```
1990 }%
```

```
1991 }
```

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

\Glsplural

```
1992 \newrobustcmd*{\Glsplural}{\@ifstar\@sGlsplural\@Glsplural}
```

Define the starred form:

```
1993 \newcommand*{\@sGlsplural}[1] []{\@Glsplural[hyper=false,#1]}
```

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

```
1994 \newcommand*{\@Glsplural}[2] []{%
```

```
1995 \new@ifnextchar[{\@Glsplural@{#1}{#2}}{\@Glsplural@{#1}{#2} []}]
```

Read in the final optional argument:

```
1996 \def\@Glsplural@#1#2[#3]{%
```

```
1997 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
1998 \protected@edef\@glo@text{\glsentryplural{#2}}%
```

Call \@gls@link

```
1999 \@gls@link[#1]{#2}{%
```

```
2000 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
```

```
2001 }%
```

```
2002 }
```

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

\GLSplural

```
2003 \newrobustcmd*{\GLSplural}{\@ifstar\@sGLSplural\@GLSplural}
```

Define the starred form:

```
2004 \newcommand*{\@GLSplural}[1] []{\@GLSplural[hyper=false,#1]}
    Defined the un-starred form. Need to determine if there is a final optional argument
2005 \newcommand*{\@GLSplural}[2] []{%
2006 \new@ifnextchar[{\@GLSplural@{#1}{#2}}{\@GLSplural@{#1}{#2} []}]
    Read in the final optional argument:
2007 \def\@GLSplural@#1#2[#3]{%
2008 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%
    Determine what the link text should be (this is stored in \@glo@text)
2009 \protected@edef\@glo@text{\glsentryplural{#2}}%
    Call \@gls@link
2010 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2011 }%
2012 }
```

`\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`

```
2013 \newrobustcmd*{\glsfirstplural}{\@ifstar\@sglsfirstplural\@glsfirstplural}
    Define the starred form:
2014 \newcommand*{\@sglsfirstplural}[1] []{\@glsfirstplural[hyper=false,#1]}
    Defined the un-starred form. Need to determine if there is a final optional argument
2015 \newcommand*{\@glsfirstplural}[2] []{%
2016 \new@ifnextchar[{\@glsfirstplural@{#1}{#2}}{\@glsfirstplural@{#1}{#2} []}]
    Read in the final optional argument:
2017 \def\@glsfirstplural@#1#2[#3]{%
2018 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%
    Determine what the link text should be (this is stored in \@glo@text)
2019 \protected@edef\@glo@text{\glsentryfirstplural{#2}}%
    Call \@gls@link
2020 \@gls@link[#1]{#2}{\@glo@text#3}%
2021 }%
2022 }
```

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

`\Glsfirstplural`

```
2023 \newrobustcmd*{\Glsfirstplural}{\@ifstar\@sGlsfirstplural\@Glsfirstplural}
    Define the starred form:
2024 \newcommand*{\@sGlsfirstplural}[1] []{\@Glsfirstplural[hyper=false,#1]}
```

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

```
2025 \newcommand*{\@Glsfirstplural}[2] []{%
2026 \new@ifnextchar [{\@Glsfirstplural@#1}{#2}]{\@Glsfirstplural@{#1}{#2} []}}
    Read in the final optional argument:
2027 \def\@Glsfirstplural@#1#2[#3]{%
2028 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%
    Determine what the link text should be (this is stored in \@glo@text)
2029 \protected@edef\@glo@text{\glsentryfirstplural{#2}}%
    Call \@gls@link
2030 \@gls@link[#1]{#2}{%
2031 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2032 }%
2033 }
```

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

`\GLSfirstplural`

```
2034 \newrobustcmd*{\@GLSfirstplural}{\@ifstar\@sGLSfirstplural\@GLSfirstplural}
    Define the starred form:
2035 \newcommand*{\@sGLSfirstplural}[1] [] {\@GLSfirstplural[hyper=false,#1]}
    Defined the un-starred form. Need to determine if there is a final optional argument
2036 \newcommand*{\@GLSfirstplural}[2] []{%
2037 \new@ifnextchar [{\@GLSfirstplural@#1}{#2}]{\@GLSfirstplural@{#1}{#2} []}}
    Read in the final optional argument:
2038 \def\@GLSfirstplural@#1#2[#3]{%
2039 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%
    Determine what the link text should be (this is stored in \@glo@text)
2040 \protected@edef\@glo@text{\glsentryfirstplural{#2}}%
    Call \@gls@link
2041 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}#3}%
2042 }%
2043 }
```

`\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`

```
2044 \newrobustcmd*{\@glsname}{\@ifstar\@sglsname\@glsname}
    Define the starred form:
2045 \newcommand*{\@sglsname}[1] [] {\@glsname[hyper=false,#1]}
```

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

```
2046 \newcommand*{\@glsname}[2] [] {%
2047 \new@ifnextchar [{\@glsname@#1}{#2}]{\@glsname@#1}{#2} []}}
```

Read in the final optional argument:

```
2048 \def\@glsname@#1#2[#3] {%
2049 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2050 \protected@edef\@glo@text{\glsentryname{#2}}%
```

Call \@gls@link

```
2051 \@gls@link[#1]{#2}{\@glo@text#3}%
2052 }%
2053 }
```

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

\Glsname

```
2054 \newrobustcmd*{\Glsname}{\@ifstar\@sGlsname\@Glsname}
```

Define the starred form:

```
2055 \newcommand*{\@sGlsname}[1] [] {\@Glsname[hyper=false,#1]}
```

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

```
2056 \newcommand*{\@Glsname}[2] [] {%
2057 \new@ifnextchar [{\@Glsname@#1}{#2}]{\@Glsname@#1}{#2} []}}
```

Read in the final optional argument:

```
2058 \def\@Glsname@#1#2[#3] {%
2059 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2060 \protected@edef\@glo@text{\glsentryname{#2}}%
```

Call \@gls@link

```
2061 \@gls@link[#1]{#2}{%
2062 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2063 }%
2064 }
```

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

\GLSname

```
2065 \newrobustcmd*{\GLSname}{\@ifstar\@sGLSname\@GLSname}
```

Define the starred form:

```
2066 \newcommand*{\@sGLSname}[1] [] {\@GLSname[hyper=false,#1]}
```

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

```
2067 \newcommand*{\@GLSname}[2] [] {%
2068 \new@ifnextchar [{\@GLSname@{#1}{#2}}{\@GLSname@{#1}{#2} []}}
```

Read in the final optional argument:

```
2069 \def\@GLSname@#1#2[#3] {%
2070 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2071 \protected@edef\@glo@text{\glsentryname{#2}}%
```

Call \@gls@link

```
2072 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2073 }%
2074 }
```

\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

```
2075 \newrobustcmd*{\glsdesc}{\@ifstar\@sglsdesc\@glsdesc}
```

Define the starred form:

```
2076 \newcommand*{\@sglsdesc}[1] [] {\@glsdesc[hyper=false,#1]}
```

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

```
2077 \newcommand*{\@glsdesc}[2] [] {%
2078 \new@ifnextchar [{\@glsdesc@{#1}{#2}}{\@glsdesc@{#1}{#2} []}}
```

Read in the final optional argument:

```
2079 \def\@glsdesc@#1#2[#3] {%
2080 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2081 \protected@edef\@glo@text{\glsentrydesc{#2}}%
```

Call \@gls@link

```
2082 \@gls@link[#1]{#2}{\@glo@text#3}%
2083 }%
2084 }
```

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

\Glsdesc

```
2085 \newrobustcmd*{\Glsdesc}{\@ifstar\@sGlsdesc\@Glsdesc}
```

Define the starred form:

```
2086 \newcommand*{\@sGlsdesc}[1] [] {\@Glsdesc[hyper=false,#1]}
```

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

```
2087 \newcommand*{\@Glsdesc}[2] [] {%
2088 \new@ifnextchar [{\@Glsdesc@{#1}{#2}}{\@Glsdesc@{#1}{#2} []}}
```

Read in the final optional argument:

```
2089 \def\@Glsdesc@#1#2[#3] {%
2090 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}
```

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

```
2091 \protected@edef\@glo@text{\glsentrydesc{#2}}%
```

Call \@gls@link

```
2092 \@gls@link[#1]{#2}{%
2093 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2094 }%
2095 }
```

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

\GLSdesc

```
2096 \newrobustcmd*{\GLSdesc}{\@ifstar\@sGLSdesc\@GLSdesc}
```

Define the starred form:

```
2097 \newcommand*{\@sGLSdesc}[1] [] {\@GLSdesc[hyper=false,#1]}
```

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

```
2098 \newcommand*{\@GLSdesc}[2] [] {%
2099 \new@ifnextchar [{\@GLSdesc@{#1}{#2}}{\@GLSdesc@{#1}{#2} []}}
```

Read in the final optional argument:

```
2100 \def\@GLSdesc@#1#2[#3] {%
2101 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}
```

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

```
2102 \protected@edef\@glo@text{\glsentrydesc{#2}}%
```

Call \@gls@link

```
2103 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}#3}%
2104 }%
2105 }
```

\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

```
2106 \newrobustcmd*{\glsdescplural}{\@ifstar\@sglsdescplural\@glsdescplural}
```

Define the starred form:

```
2107 \newcommand*{\@sglsdescplural}[1] [] {\@glsdescplural[hyper=false,#1]}
```

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

```
2108 \newcommand*{\glsdescplural}[2] []{%  
2109 \new@ifnextchar[{\@glsdescplural@{#1}{#2}}{\@glsdescplural@{#1}{#2}[]}}
```

Read in the final optional argument:

```
2110 \def\@glsdescplural@#1#2[#3]{%  
2111 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2112 \protected@edef\@glo@text{\glsentrydescplural{#2}}%
```

Call \@gls@link

```
2113 \@gls@link[#1]{#2}{\@glo@text#3}%  
2114 }%  
2115 }
```

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

\Glsdescplural

```
2116 \newrobustcmd*{\Glsdescplural}{\@ifstar\@sGlsdescplural\@Glsdescplural}
```

Define the starred form:

```
2117 \newcommand*{\@sGlsdescplural}[1] []{\@Glsdescplural[hyper=false,#1]}
```

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

```
2118 \newcommand*{\@Glsdescplural}[2] []{%  
2119 \new@ifnextchar[{\@Glsdescplural@{#1}{#2}}{\@Glsdescplural@{#1}{#2}[]}}
```

Read in the final optional argument:

```
2120 \def\@Glsdescplural@#1#2[#3]{%  
2121 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2122 \protected@edef\@glo@text{\glsentrydescplural{#2}}%
```

Call \@gls@link

```
2123 \@gls@link[#1]{#2}{%  
2124 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%  
2125 }%  
2126 }
```

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

\GLSdescplural

```
2127 \newrobustcmd*{\GLSdescplural}{\@ifstar\@sGLSdescplural\@GLSdescplural}
```

Define the starred form:

```
2128 \newcommand*{\@sGLSdescplural}[1] []{\@GLSdescplural[hyper=false,#1]}
```

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

```
2129 \newcommand*{\GLSdescplural}[2] [] {%
2130 \new@ifnextchar [{\@GLSdescplural@{#1}{#2}}{\@GLSdescplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
2131 \def\@GLSdescplural@#1#2[#3] {%
2132 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%
```

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

```
2133 \protected@edef\@glo@text{\glsentrydescplural{#2}}%
```

Call \@gls@link

```
2134 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2135 }%
2136 }
```

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

\glsymbol

```
2137 \newrobustcmd*{\glsymbol}{\@ifstar\@sglsymbol\@glsymbol}
```

Define the starred form:

```
2138 \newcommand*{\@sglsymbol}[1] [] {\@glsymbol[hyper=false,#1]}
```

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

```
2139 \newcommand*{\@glsymbol}[2] [] {%
2140 \new@ifnextchar [{\@glsymbol@{#1}{#2}}{\@glsymbol@{#1}{#2} []}]}
```

Read in the final optional argument:

```
2141 \def\@glsymbol@#1#2[#3] {%
2142 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%
```

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

```
2143 \protected@edef\@glo@text{\glsentrysymbol{#2}}%
```

Call \@gls@link

```
2144 \@gls@link[#1]{#2}{\@glo@text#3}%
2145 }%
2146 }
```

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

\Glsymbol

```
2147 \newrobustcmd*{\Glsymbol}{\@ifstar\@sGlsymbol\@Glsymbol}
```

Define the starred form:

```
2148 \newcommand*{\@sGlsymbol}[1] [] {\@Glsymbol[hyper=false,#1]}
```

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

```
2149 \newcommand*{\@Glssymbol}[2] [] {%
2150 \new@ifnextchar [{\@Glssymbol@{#1}{#2}}{\@Glssymbol@{#1}{#2} []}]
    Read in the final optional argument:
2151 \def\@Glssymbol@#1#2[#3] {%
2152 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%
    Determine what the link text should be (this is stored in \@glo@text)
2153 \protected@edef\@glo@text{\glsentrysymbol{#2}}%
    Call \@gls@link
2154 \@gls@link[#1]{#2}{%
2155     \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2156 }%
2157 }
```

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

\GLSsymbol

```
2158 \newrobustcmd*{\GLSsymbol}{\@ifstar\@sGLSsymbol\@GLSsymbol}
```

Define the starred form:

```
2159 \newcommand*{\@sGLSsymbol}[1] [] {\@GLSsymbol[hyper=false,#1]}
```

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

```
2160 \newcommand*{\@GLSsymbol}[2] [] {%
2161 \new@ifnextchar [{\@GLSsymbol@{#1}{#2}}{\@GLSsymbol@{#1}{#2} []}]
    Read in the final optional argument:
2162 \def\@GLSsymbol@#1#2[#3] {%
2163 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%
    Determine what the link text should be (this is stored in \@glo@text)
2164 \protected@edef\@glo@text{\glsentrysymbol{#2}}%
    Call \@gls@link
2165 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}#3}%
2166 }%
2167 }
```

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

\glsymbolplural

```
2168 \newrobustcmd*{\glsymbolplural}{\@ifstar\@sglsymbolplural\@glsymbolplural}
```

Define the starred form:

```
2169 \newcommand*{\@sglsymbolplural}[1] [] {\@glsymbolplural[hyper=false,#1]}
```

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

```
2170 \newcommand*{\glssymbolplural}[2] [] {%
2171 \new@ifnextchar [{\@glssymbolplural@{#1}{#2}}{\@glssymbolplural@{#1}{#2} [] }}
```

Read in the final optional argument:

```
2172 \def\@glssymbolplural@#1#2[#3] {%
2173 \glsdoifexists{#2}{\edef\@glo@type{\glstrytype{#2}}%
```

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

```
2174 \protected@edef\@glo@text{\glstrytype{#2}}%
```

Call \@gls@link

```
2175 \@gls@link[#1]{#2}{\@glo@text#3}%
2176 }%
2177 }
```

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

\Glsymbolplural

```
2178 \newrobustcmd*{\Glsymbolplural}{\@ifstar\@sGlsymbolplural\@Glsymbolplural}
```

Define the starred form:

```
2179 \newcommand*{\@sGlsymbolplural}[1] [] {\@Glsymbolplural[hyper=false,#1]}
```

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

```
2180 \newcommand*{\@Glsymbolplural}[2] [] {%
2181 \new@ifnextchar [{\@Glsymbolplural@{#1}{#2}}{\@Glsymbolplural@{#1}{#2} [] }}
```

Read in the final optional argument:

```
2182 \def\@Glsymbolplural@#1#2[#3] {%
2183 \glsdoifexists{#2}{\edef\@glo@type{\glstrytype{#2}}%
```

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

```
2184 \protected@edef\@glo@text{\glstrytype{#2}}%
```

Call \@gls@link

```
2185 \@gls@link[#1]{#2}{%
2186 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2187 }%
2188 }
```

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

\GLSsymbolplural

```
2189 \newrobustcmd*{\GLSsymbolplural}{\@ifstar\@sGLSsymbolplural\@GLSsymbolplural}
```

Define the starred form:

```
2190 \newcommand*{\@sGLSsymbolplural}[1] [] {\@GLSsymbolplural[hyper=false,#1]}
```

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

```
2191 \newcommand*{\GLSsymbolplural}[2] [] {%
2192 \new@ifnextchar[{\@GLSsymbolplural@{#1}{#2}}{\@GLSsymbolplural@{#1}{#2} [] }}
    Read in the final optional argument:
2193 \def\@GLSsymbolplural@#1#2[#3] {%
2194 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%
    Determine what the link text should be (this is stored in \@glo@text)
2195 \protected@edef\@glo@text{\glsentrysymbolplural{#2}}%
    Call \@gls@link
2196 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2197 }%
2198 }
```

`\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`

```
2199 \newrobustcmd*{\glsuseri}{\@ifstar\@sglsuseri\@glsuseri}
```

Define the starred form:

```
2200 \newcommand*{\@sglsuseri}[1] [] {\@glsuseri[hyper=false,#1]}
```

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

```
2201 \newcommand*{\@glsuseri}[2] [] {%
2202 \new@ifnextchar[{\@glsuseri@{#1}{#2}}{\@glsuseri@{#1}{#2} [] }}
    Read in the final optional argument:
```

```
2203 \def\@glsuseri@#1#2[#3] {%
2204 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%
    Determine what the link text should be (this is stored in \@glo@text)
```

```
2205 \protected@edef\@glo@text{\glsentryuseri{#2}}%
```

Call `\@gls@link`

```
2206 \@gls@link[#1]{#2}{\@glo@text#3}%
2207 }%
2208 }
```

`\Glsuseri` behaves like `\glsuseri` except that the first letter is converted to uppercase.

`\Glsuseri`

```
2209 \newrobustcmd*{\Glsuseri}{\@ifstar\@sGlsuseri\@Glsuseri}
```

Define the starred form:

```
2210 \newcommand*{\@sGlsuseri}[1] [] {\@Glsuseri[hyper=false,#1]}
```

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

```
2211 \newcommand*{\@Glsuseri}[2] [] {%  
2212 \new@ifnextchar [{\@Glsuseri@{#1}{#2}}{\@Glsuseri@{#1}{#2} []}] }
```

Read in the final optional argument:

```
2213 \def\@Glsuseri@#1#2[#3] {%  
2214 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%
```

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

```
2215 \protected@edef\@glo@text{\glsentryuseri{#2}}%
```

Call \@gls@link

```
2216 \@gls@link[#1]{#2}{%  
2217 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%  
2218 }%  
2219 }
```

\GLSuseri behaves like \glsuseri except that the link text is converted to uppercase.

\GLSuseri

```
2220 \newrobustcmd*{\GLSuseri}{\@ifstar\@sGLSuseri\@GLSuseri}
```

Define the starred form:

```
2221 \newcommand*{\@sGLSuseri}[1] [] {\@GLSuseri[hyper=false,#1]}
```

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

```
2222 \newcommand*{\@GLSuseri}[2] [] {%  
2223 \new@ifnextchar [{\@GLSuseri@{#1}{#2}}{\@GLSuseri@{#1}{#2} []}] }
```

Read in the final optional argument:

```
2224 \def\@GLSuseri@#1#2[#3] {%  
2225 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%
```

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

```
2226 \protected@edef\@glo@text{\glsentryuseri{#2}}%
```

Call \@gls@link

```
2227 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}#3}%  
2228 }%  
2229 }
```

\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

```
2230 \newrobustcmd*{\glsuserii}{\@ifstar\@sglsuserii\@glsuserii}
```

Define the starred form:

```
2231 \newcommand*{\@sglsuserii}[1] [] {\@glsuserii[hyper=false,#1]}
```

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

```
2232 \newcommand*{\glsuserii}[2] [] {%
2233 \new@ifnextchar[{\@glsuserii@{#1}{#2}}{\@glsuserii@{#1}{#2} []]}
    Read in the final optional argument:
2234 \def\@glsuserii@#1#2[#3] {%
2235 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%
    Determine what the link text should be (this is stored in \@glo@text)
2236 \protected@edef\@glo@text{\glsentryuserii{#2}}%
    Call \@gls@link
2237 \@gls@link[#1]{#2}{\@glo@text#3}%
2238 }%
2239 }
```

`\Glsuserii` behaves like `\glsuserii` except that the first letter is converted to uppercase.

`\Glsuserii`

```
2240 \newrobustcmd*{\Glsuserii}{\@ifstar\@sGlsuserii\@Glsuserii}
```

Define the starred form:

```
2241 \newcommand*{\@sGlsuserii}[1] [] {\@Glsuserii[hyper=false,#1]}
    Defined the un-starred form. Need to determine if there is a final optional argument
2242 \newcommand*{\@Glsuserii}[2] [] {%
2243 \new@ifnextchar[{\@Glsuserii@{#1}{#2}}{\@Glsuserii@{#1}{#2} []]}
    Read in the final optional argument:
2244 \def\@Glsuserii@#1#2[#3] {%
2245 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%
    Determine what the link text should be (this is stored in \@glo@text)
2246 \protected@edef\@glo@text{\glsentryuserii{#2}}%
    Call \@gls@link
2247 \@gls@link[#1]{#2}{%
2248 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2249 }%
2250 }
```

`\GLSuserii` behaves like `\glsuserii` except that the link text is converted to uppercase.

`\GLSuserii`

```
2251 \newrobustcmd*{\GLSuserii}{\@ifstar\@sGLSuserii\@GLSuserii}
```

Define the starred form:

```
2252 \newcommand*{\@sGLSuserii}[1] [] {\@GLSuserii[hyper=false,#1]}
```

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

```
2253 \newcommand*{\GLSuserii}[2] [] {%  
2254 \new@ifnextchar [{\@GLSuserii@{#1}{#2}}{\@GLSuserii@{#1}{#2} []}]
```

Read in the final optional argument:

```
2255 \def\@GLSuserii@#1#2[#3] {%  
2256 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}
```

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

```
2257 \protected@edef\@glo@text{\glsentryuserii{#2}}%
```

Call \@gls@link

```
2258 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%  
2259 }%  
2260 }
```

\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

```
2261 \newrobustcmd*{\glsuseriii}{\@ifstar\@sglsuseriii\@glsuseriii}
```

Define the starred form:

```
2262 \newcommand*{\@sglsuseriii}[1] [] {\@glsuseriii[hyper=false,#1]}
```

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

```
2263 \newcommand*{\@glsuseriii}[2] [] {%  
2264 \new@ifnextchar [{\@glsuseriii@{#1}{#2}}{\@glsuseriii@{#1}{#2} []}]
```

Read in the final optional argument:

```
2265 \def\@glsuseriii@#1#2[#3] {%  
2266 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}
```

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

```
2267 \protected@edef\@glo@text{\glsentryuseriii{#2}}%
```

Call \@gls@link

```
2268 \@gls@link[#1]{#2}{\@glo@text#3}}%  
2269 }%  
2270 }
```

\Glsuseriii behaves like \glsuseriii except that the first letter is converted to uppercase.

\Glsuseriii

```
2271 \newrobustcmd*{\Glsuseriii}{\@ifstar\@sGlsuseriii\@Glsuseriii}
```

Define the starred form:

```
2272 \newcommand*{\@sGlsuseriii}[1] [] {\@Glsuseriii[hyper=false,#1]}
```

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

```
2273 \newcommand*{\@Glsuseriii}[2] [] {%
2274 \new@ifnextchar [ {\@Glsuseriii@{#1}{#2}} {\@Glsuseriii@{#1}{#2} [] ] }
```

Read in the final optional argument:

```
2275 \def \@Glsuseriii@#1#2[#3] {%
2276 \glsdoifexists{#2} {\edef \@glo@type {\glsentrytype{#2}} }
```

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

```
2277 \protected@edef \@glo@text {\glsentryuseriii{#2}} %
```

Call \@gls@link

```
2278 \@gls@link[#1]{#2} {%
2279 \expandafter \makefirstuc \expandafter {\@glo@text} #3 %
2280 } %
2281 }
```

\GLSuseriii behaves like \glsuseriii except that the link text is converted to uppercase.

\GLSuseriii

```
2282 \newrobustcmd*{\GLSuseriii} {\@ifstar \@sGLSuseriii \@GLSuseriii}
```

Define the starred form:

```
2283 \newcommand*{\@sGLSuseriii}[1] [] {\@GLSuseriii[hyper=false,#1]}
```

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

```
2284 \newcommand*{\@GLSuseriii}[2] [] {%
2285 \new@ifnextchar [ {\@GLSuseriii@{#1}{#2}} {\@GLSuseriii@{#1}{#2} [] ] }
```

Read in the final optional argument:

```
2286 \def \@GLSuseriii@#1#2[#3] {%
2287 \glsdoifexists{#2} {\edef \@glo@type {\glsentrytype{#2}} }
```

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

```
2288 \protected@edef \@glo@text {\glsentryuseriii{#2}} %
```

Call \@gls@link

```
2289 \@gls@link[#1]{#2} {\MakeUppercase {\@glo@text} #3} %
2290 } %
2291 }
```

\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

```
2292 \newrobustcmd*{\glsuseriv} {\@ifstar \@sglsuseriv \@glsuseriv}
```

Define the starred form:

```
2293 \newcommand*{\@sglsuseriv}[1] [] {\@glsuseriv[hyper=false,#1]}
```

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

```
2294 \newcommand*{\@glsuseriv}[2] [] {%  
2295 \new@ifnextchar [{\@glsuseriv@{#1}{#2}}{\@glsuseriv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
2296 \def\@glsuseriv@#1#2[#3] {%  
2297 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}
```

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

```
2298 \protected@edef\@glo@text{\glsentryuseriv{#2}}%
```

Call \@gls@link

```
2299 \@gls@link[#1]{#2}{\@glo@text#3}%  
2300 }%  
2301 }
```

\Glsuseriv behaves like \glsuseriv except that the first letter is converted to uppercase.

\Glsuseriv

```
2302 \newrobustcmd*{\Glsuseriv}{\@ifstar\@sGlsuseriv\@Glsuseriv}
```

Define the starred form:

```
2303 \newcommand*{\@sGlsuseriv}[1] [] {\@Glsuseriv[hyper=false,#1]}
```

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

```
2304 \newcommand*{\@Glsuseriv}[2] [] {%  
2305 \new@ifnextchar [{\@Glsuseriv@{#1}{#2}}{\@Glsuseriv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
2306 \def\@Glsuseriv@#1#2[#3] {%  
2307 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}
```

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

```
2308 \protected@edef\@glo@text{\glsentryuseriv{#2}}%
```

Call \@gls@link

```
2309 \@gls@link[#1]{#2}{%  
2310 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%  
2311 }%  
2312 }
```

\GLSuseriv behaves like \glsuseriv except that the link text is converted to uppercase.

\GLSuseriv

```
2313 \newrobustcmd*{\GLSuseriv}{\@ifstar\@sGLSuseriv\@GLSuseriv}
```

Define the starred form:

```
2314 \newcommand*{\@sGLSuseriv}[1] [] {\@GLSuseriv[hyper=false,#1]}
```

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

```
2315 \newcommand*{\GLSuseriv}[2] [] {%
2316 \new@ifnextchar[{\@GLSuseriv@{#1}{#2}}{\@GLSuseriv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
2317 \def\@GLSuseriv@#1#2[#3] {%
2318 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2319 \protected@edef\@glo@text{\glsentryuseriv{#2}}%
```

Call \@gls@link

```
2320 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2321 }%
2322 }
```

\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

```
2323 \newrobustcmd*{\glsuserv}{\@ifstar\@sglsuserv\@glsuserv}
```

Define the starred form:

```
2324 \newcommand*{\@sglsuserv}[1] [] {\@glsuserv[hyper=false,#1]}
```

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

```
2325 \newcommand*{\@glsuserv}[2] [] {%
2326 \new@ifnextchar[{\@glsuserv@{#1}{#2}}{\@glsuserv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
2327 \def\@glsuserv@#1#2[#3] {%
2328 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2329 \protected@edef\@glo@text{\glsentryuseriv{#2}}%
```

Call \@gls@link

```
2330 \@gls@link[#1]{#2}{\@glo@text#3}%
2331 }%
2332 }
```

\Glsuserv behaves like \glsuserv except that the first letter is converted to uppercase.

\Glsuserv

```
2333 \newrobustcmd*{\Glsuserv}{\@ifstar\@sGlsuserv\@Glsuserv}
```

Define the starred form:

```
2334 \newcommand*{\@sGlsuserv}[1] [] {\@Glsuserv[hyper=false,#1]}
```

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

```
2335 \newcommand*{\@Glsuserv}[2] [] {%  
2336 \new@ifnextchar [{\@Glsuserv@{#1}{#2}}{\@Glsuserv@{#1}{#2} []}] }
```

Read in the final optional argument:

```
2337 \def\@Glsuserv@#1#2[#3] {%  
2338 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%
```

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

```
2339 \protected@edef\@glo@text{\glsentryuserv{#2}}%
```

Call \@gls@link

```
2340 \@gls@link[#1]{#2}{%  
2341 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%  
2342 }%  
2343 }
```

\GLSuserv behaves like \glsuserv except that the link text is converted to uppercase.

\GLSuserv

```
2344 \newrobustcmd*{\@ifstar\@sGLSuserv\@GLSuserv}
```

Define the starred form:

```
2345 \newcommand*{\@sGLSuserv}[1] [] {\@GLSuserv[hyper=false,#1]}
```

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

```
2346 \newcommand*{\@GLSuserv}[2] [] {%  
2347 \new@ifnextchar [{\@GLSuserv@{#1}{#2}}{\@GLSuserv@{#1}{#2} []}] }
```

Read in the final optional argument:

```
2348 \def\@GLSuserv@#1#2[#3] {%  
2349 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%
```

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

```
2350 \protected@edef\@glo@text{\glsentryuserv{#2}}%
```

Call \@gls@link

```
2351 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}#3}%  
2352 }%  
2353 }
```

\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

```
2354 \newrobustcmd*{\@ifstar\@sglsuservi\@glsuservi}
```

Define the starred form:

```
2355 \newcommand*{\@sglsuservi}[1] [] {\@glsuservi[hyper=false,#1]}
```

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

```
2356 \newcommand*{\@glsuservi}[2] [] {%
2357 \new@ifnextchar[{\@glsuservi@#1}{#2}]{\@glsuservi@{#1}{#2} []}}
```

Read in the final optional argument:

```
2358 \def\@glsuservi@#1#2[#3] {%
2359 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}
```

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

```
2360 \protected@edef\@glo@text{\glsentryuservi{#2}}%
```

Call \@gls@link

```
2361 \@gls@link[#1]{#2}{\@glo@text#3}%
2362 }%
2363 }
```

\Glsuservi behaves like \glsuservi except that the first letter is converted to uppercase.

\Glsuservi

```
2364 \newrobustcmd*{\Glsuservi}{\@ifstar\@sGlsuservi\@Glsuservi}
```

Define the starred form:

```
2365 \newcommand*{\@sGlsuservi}[1] [] {\@Glsuservi[hyper=false,#1]}
```

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

```
2366 \newcommand*{\@Glsuservi}[2] [] {%
2367 \new@ifnextchar[{\@Glsuservi@#1}{#2}]{\@Glsuservi@{#1}{#2} []}}
```

Read in the final optional argument:

```
2368 \def\@Glsuservi@#1#2[#3] {%
2369 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}
```

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

```
2370 \protected@edef\@glo@text{\glsentryuservi{#2}}%
```

Call \@gls@link

```
2371 \@gls@link[#1]{#2}{%
2372 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2373 }%
2374 }
```

\GLSuservi behaves like \glsuservi except that the link text is converted to uppercase.

\GLSuservi

```
2375 \newrobustcmd*{\GLSuservi}{\@ifstar\@sGLSuservi\@GLSuservi}
```

Define the starred form:

```
2376 \newcommand*{\@sGLSuservi}[1] [] {\@GLSuservi[hyper=false,#1]}
```

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

```
2377 \newcommand*{\GLSuservi}[2] [] {%
2378 \new@ifnextchar[{\@GLSuservi@{#1}{#2}}{\@GLSuservi@{#1}{#2} []}]
```

Read in the final optional argument:

```
2379 \def\@GLSuservi@#1#2[#3] {%
2380 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}
```

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

```
2381 \protected@edef\@glo@text{\glsentryuseri{#2}}%
```

Call \@gls@link

```
2382 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2383 }%
2384 }
```

Now deal with acronym related keys. First the short form:

`\acrshort`

```
2385 \newrobustcmd*{\acrshort}{\@ifstar\s@acrshort\ns@acrshort}
```

Define the starred form:

```
2386 \newcommand*{\s@acrshort}[2] [] {%
2387 \new@ifnextchar[{\@acrshort{hyper=false,#1}{#2}}%
2388 \@acrshort{hyper=false,#1}{#2} []}%
2389 }
```

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

```
2390 \newcommand*{\ns@acrshort}[2] [] {%
2391 \new@ifnextchar[{\@acrshort{#1}{#2}}{\@acrshort{#1}{#2} []}]%
2392 }
```

Read in the final optional argument:

```
2393 \def\@acrshort#1#2[#3] {%
2394 \glsdoifexists{#2}%
2395 {%
2396 \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2397 \protected@edef\@glo@text{\glsentryshort{#2}}%
```

Call \@gls@link

```
2398 \@gls@link[#1]{#2}{\acronymfont{\@glo@text#3}}%
2399 }%
2400 }
```

`\Acrshort`

```
2401 \newrobustcmd*{\Acrshort}{\@ifstar\s@Acrshort\ns@Acrshort}
```

Define the starred form:

```
2402 \newcommand*{\s@Acrshort}[2] [] {%
2403   \new@ifnextchar[{\@Acrshort{hyper=false,#1}{#2}}%
2404     {\@Acrshort{hyper=false,#1}{#2} []}%
2405 }
```

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

```
2406 \newcommand*{\ns@Acrshort}[2] [] {%
2407   \new@ifnextchar[{\@Acrshort{#1}{#2}}{\@Acrshort{#1}{#2} []}%
2408 }
```

Read in the final optional argument:

```
2409 \def\@Acrshort#1#2[#3] {%
2410   \glsdoifexists{#2}%
2411   {%
2412     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2413   \protected@edef\@glo@text{\glsentryshort{#2}}%
```

Call \@gls@link

```
2414   \@gls@link[#1]{#2}%
2415   {%
2416     \acronymfont{\expandafter\makefirstuc\expandafter{\@glo@text}}#3%
2417   }%
2418 }%
2419 }
```

\ACRshort

```
2420 \newrobustcmd*{\ACRshort}{\@ifstar\s@ACRshort\ns@ACRshort}
```

Define the starred form:

```
2421 \newcommand*{\s@ACRshort}[2] [] {%
2422   \new@ifnextchar[{\@ACRshort{hyper=false,#1}{#2}}%
2423     {\@ACRshort{hyper=false,#1}{#2} []}%
2424 }
```

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

```
2425 \newcommand*{\ns@ACRshort}[2] [] {%
2426   \new@ifnextchar[{\@ACRshort{#1}{#2}}{\@ACRshort{#1}{#2} []}%
2427 }
```

Read in the final optional argument:

```
2428 \def\@ACRshort#1#2[#3] {%
2429   \glsdoifexists{#2}%
2430   {%
2431     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2432   \protected@edef\@glo@text{\glsentryshort{#2}}%
```

Call \@gls@link

```
2433   \@gls@link[#1]{#2}{\acronymfont{\MakeUppercase{\@glo@text#3}}}%  
2434   }%  
2435 }
```

Short plural:

\acrshortpl

```
2436 \newrobustcmd*{\acrshortpl}{\@ifstar\s@acrshortpl\ns@acrshortpl}
```

Define the starred form:

```
2437 \newcommand*{\s@acrshortpl}[2] [] {%  
2438   \new@ifnextchar[{\@acrshortpl{hyper=false,#1}{#2}}%  
2439     {\@acrshortpl{hyper=false,#1}{#2} []}%  
2440 }
```

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

```
2441 \newcommand*{\ns@acrshortpl}[2] [] {%  
2442   \new@ifnextchar[{\@acrshortpl{#1}{#2}}{\@acrshortpl{#1}{#2} []}%  
2443 }
```

Read in the final optional argument:

```
2444 \def\@acrshortpl#1#2[#3] {%  
2445   \glsdoifexists{#2}%  
2446   {%  
2447     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2448   \protected@edef\@glo@text{\glsentryshortpl{#2}}%
```

Call \@gls@link

```
2449   \@gls@link[#1]{#2}{\acronymfont{\@glo@text#3}}%  
2450   }%  
2451 }
```

\Acrshortpl

```
2452 \newrobustcmd*{\Acrshortpl}{\@ifstar\s@Acrshortpl\ns@Acrshortpl}
```

Define the starred form:

```
2453 \newcommand*{\s@Acrshortpl}[2] [] {%  
2454   \new@ifnextchar[{\@Acrshortpl{hyper=false,#1}{#2}}%  
2455     {\@Acrshortpl{hyper=false,#1}{#2} []}%  
2456 }
```

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

```
2457 \newcommand*{\ns@Acrshortpl}[2] [] {%  
2458   \new@ifnextchar[{\@Acrshortpl{#1}{#2}}{\@Acrshortpl{#1}{#2} []}%  
2459 }
```

Read in the final optional argument:

```
2460 \def\@Acrshortpl#1#2[#3]{%
2461   \glsdoifexists{#2}%
2462   {%
2463     \edef\@glo@type{\glsentrytype{#2}}%
      Determine what the link text should be (this is stored in \@glo@text)
2464     \protected@edef\@glo@text{\glsentryshortpl{#2}}%
      Call \@gls@link
2465     \@gls@link[#1]{#2}%
2466     {%
2467       \acronymfont{\expandafter\makefirstuc\expandafter{\@glo@text}}#3%
2468     }%
2469   }%
2470 }
```

`\ACRshortpl`

```
2471 \newrobustcmd*{\ACRshortpl}{\@ifstar\s@ACRshortpl\ns@ACRshortpl}
```

Define the starred form:

```
2472 \newcommand*{\s@ACRshortpl}[2][ ]{%
2473   \new@ifnextchar[{\@ACRshortpl{hyper=false,#1}{#2}}%
2474     {\@ACRshortpl{hyper=false,#1}{#2}[ ]}%
2475 }
```

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

```
2476 \newcommand*{\ns@ACRshortpl}[2][ ]{%
2477   \new@ifnextchar[{\@ACRshortpl{#1}{#2}}{\@ACRshortpl{#1}{#2}[ ]}%
2478 }
```

Read in the final optional argument:

```
2479 \def\@ACRshortpl#1#2[#3]{%
2480   \glsdoifexists{#2}%
2481   {%
2482     \edef\@glo@type{\glsentrytype{#2}}%
      Determine what the link text should be (this is stored in \@glo@text)
2483     \protected@edef\@glo@text{\glsentryshortpl{#2}}%
      Call \@gls@link
2484     \@gls@link[#1]{#2}{\acronymfont{\MakeUppercase{\@glo@text#3}}}%
2485   }%
2486 }
```

`\acrlong`

```
2487 \newrobustcmd*{\acrlong}{\@ifstar\s@acrlong\ns@acrlong}
```

Define the starred form:

```
2488 \newcommand*{\s@acrlong}[2] [] {%
2489   \new@ifnextchar[{\@acrlong{hyper=false,#1}{#2}}{%
2490     {\@acrlong{hyper=false,#1}{#2} []}%
2491 }
```

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

```
2492 \newcommand*{\ns@acrlong}[2] [] {%
2493   \new@ifnextchar[{\@acrlong{#1}{#2}}{\@acrlong{#1}{#2} []}%
2494 }
```

Read in the final optional argument:

```
2495 \def\@acrlong#1#2[#3] {%
2496   \glsdoifexists{#2}%
2497   {%
2498     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2499   \protected@edef\@glo@text{\glsentrylong{#2}}%
```

Call \@gls@link

```
2500   \@gls@link[#1]{#2}{\@glo@text#3}%
2501 }%
2502 }
```

\Acrlong

```
2503 \newrobustcmd*{\Acrlong}{\@ifstar\s@Acrlong\ns@Acrlong}
```

Define the starred form:

```
2504 \newcommand*{\s@Acrlong}[2] [] {%
2505   \new@ifnextchar[{\@Acrlong{hyper=false,#1}{#2}}{%
2506     {\@Acrlong{hyper=false,#1}{#2} []}%
2507 }
```

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

```
2508 \newcommand*{\ns@Acrlong}[2] [] {%
2509   \new@ifnextchar[{\@Acrlong{#1}{#2}}{\@Acrlong{#1}{#2} []}%
2510 }
```

Read in the final optional argument:

```
2511 \def\@Acrlong#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{\glsentrylong{#2}}%
```

```

Call \@gls@link
2516   \@gls@link[#1]{#2}%
2517   {%
2518     \expandafter\makefirstuc\expandafter{\@glo@text}#3%
2519   }%
2520 }%
2521 }

\ACRlong
2522 \newrobustcmd*{\ACRlong}{\@ifstar\s@ACRlong\ns@ACRlong}

Define the starred form:
2523 \newcommand*{\s@ACRlong}[2] [] {%
2524   \new@ifnextchar[{\@ACRlong{hyper=false,#1}{#2}}%
2525     {\@ACRlong{hyper=false,#1}{#2} []}%
2526 }

Defined the un-starred form. Need to determine if there is a final optional argu-
ment
2527 \newcommand*{\ns@ACRlong}[2] [] {%
2528   \new@ifnextchar[{\@ACRlong{#1}{#2}}{\@ACRlong{#1}{#2} []}%
2529 }

Read in the final optional argument:
2530 \def\@ACRlong#1#2[#3]{%
2531   \glsdoifexists{#2}%
2532   {%
2533     \edef\@glo@type{\glsentrytype{#2}}%
2534     \protected@edef\@glo@text{\glsentrylong{#2}}%
2535     Call \@gls@link
2536     \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2537   }%
2538   \protected@edef\@glo@text{\glsentrylong{#2}}%
2539   Call \@gls@link
2540   \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2541 }%
2542 }

Short plural:

\acrlongpl
2543 \newrobustcmd*{\acrlongpl}{\@ifstar\s@acrlongpl\ns@acrlongpl}

Define the starred form:
2539 \newcommand*{\s@acrlongpl}[2] [] {%
2540   \new@ifnextchar[{\@acrlongpl{hyper=false,#1}{#2}}%
2541     {\@acrlongpl{hyper=false,#1}{#2} []}%
2542 }

Defined the un-starred form. Need to determine if there is a final optional argu-
ment
2543 \newcommand*{\ns@acrlongpl}[2] [] {%
2544   \new@ifnextchar[{\@acrlongpl{#1}{#2}}{\@acrlongpl{#1}{#2} []}%
2545 }

```

Read in the final optional argument:

```
2546 \def\acrlongpl#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{\glsentrylongpl{#2}}%
      Call \@gls@link
2551     \@gls@link[#1]{#2}{\@glo@text#3}%
2552   }%
2553 }
```

\Acrlongpl

```
2554 \newrobustcmd*{\Acrlongpl}{\@ifstar\s@Acrlongpl\ns@Acrlongpl}
```

Define the starred form:

```
2555 \newcommand*{\s@Acrlongpl}[2] []{%
2556   \new@ifnextchar[{\@Acrlongpl{hyper=false#1}{#2}}%
2557     {\@Acrlongpl{hyper=false,#1}{#2} []}%
2558 }
```

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

```
2559 \newcommand*{\ns@Acrlongpl}[2] []{%
2560   \new@ifnextchar[{\@Acrlongpl{#1}{#2}}{\@Acrlongpl{#1}{#2} []}%
2561 }
```

Read in the final optional argument:

```
2562 \def\Acrlongpl#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{\glsentrylongpl{#2}}%
      Call \@gls@link
2567     \@gls@link[#1]{#2}%
2568     {%
2569       \expandafter\makefirstuc\expandafter{\@glo@text}#3%
2570     }%
2571   }%
2572 }
```

\ACRlongpl

```
2573 \newrobustcmd*{\ACRlongpl}{\@ifstar\s@ACRlongpl\ns@ACRlongpl}
```

Define the starred form:

```
2574 \newcommand*{\s@ACRlongpl}[2] [] {%
2575   \new@ifnextchar[{\@ACRlongpl{hyper=false,#1}{#2}}%
2576     {\@ACRlongpl{hyper=false,#1}{#2} []}%
2577 }
```

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

```
2578 \newcommand*{\ns@ACRlongpl}[2] [] {%
2579   \new@ifnextchar[{\@ACRlongpl{#1}{#2}}{\@ACRlongpl{#1}{#2} []}%
2580 }
```

Read in the final optional argument:

```
2581 \def\@ACRlongpl#1#2[#3] {%
2582   \glsdoifexists{#2}%
2583   {%
2584     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2585   \protected@edef\@glo@text{\glsentrylongpl{#2}}%
```

Call \@gls@link

```
2586   \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2587   }%
2588 }
```

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`

```
2589 \newcommand*{\glsentryname}[1]{\csname glo@#1@name\endcsname}
```

`\Glsentryname`

```
2590 \newcommand*{\Glsentryname}[1] {%
2591   \protected@edef\@glo@text{\csname glo@#1@name\endcsname}%
2592   \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`

```
2593 \newcommand*{\glsentrydesc}[1]{\csname glo@#1@desc\endcsname}
```

`\Glsentrydesc`

```
2594 \newcommand*{\Glsentrydesc}[1]{%
2595 \protected@edef\@glo@text{\csname glo@#1@desc\endcsname}%
2596 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Plural form:

`\glsentrydescplural`

```
2597 \newcommand*{\glsentrydescplural}[1]{%
2598 \csname glo@#1@descplural\endcsname}
```

`\Glsentrydescplural`

```
2599 \newcommand*{\Glsentrydescplural}[1]{%
2600 \protected@edef\@glo@text{\csname glo@#1@descplural\endcsname}%
2601 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Get the entry text, as specified by the text key when the entry was defined.
The argument is the label associated with the entry:

`\glsentrytext`

```
2602 \newcommand*{\glsentrytext}[1]{\csname glo@#1@text\endcsname}
```

`\Glsentrytext`

```
2603 \newcommand*{\Glsentrytext}[1]{%
2604 \protected@edef\@glo@text{\csname glo@#1@text\endcsname}%
2605 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Get the plural form:

`\glsentryplural`

```
2606 \newcommand*{\glsentryplural}[1]{\csname glo@#1@plural\endcsname}
```

`\Glsentryplural`

```
2607 \newcommand*{\Glsentryplural}[1]{%
2608 \protected@edef\@glo@text{\csname glo@#1@plural\endcsname}%
2609 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Get the symbol associated with this entry. The argument is the label associated with the entry. Note that unless you used `symbol=false` in the `sanitize` package option you may get unexpected results if the symbol key contained any commands.

`\glsentrysymbol`

```
2610 \newcommand*{\glsentrysymbol}[1]{\csname glo@#1@symbol\endcsname}
```

`\Glsentrysymbol`

```
2611 \newcommand*{\Glsentrysymbol}[1]{%
2612 \protected@edef\@glo@text{\csname glo@#1@symbol\endcsname}%
2613 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Plural form:

`\glsentrysymbolplural`

```
2614 \newcommand*{\glsentrysymbolplural}[1]{%
2615 \csname glo@#1@symbolplural\endcsname}
```

`\Glsentrysymbolplural`

```
2616 \newcommand*{\Glsentrysymbolplural}[1]{%
2617 \protected@edef\@glo@text{\csname glo@#1@symbolplural\endcsname}%
2618 \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`

```
2619 \newcommand*{\glsentryfirst}[1]{\csname glo@#1@first\endcsname}
```

`\Glsentryfirst`

```
2620 \newcommand*{\Glsentryfirst}[1]{%
2621 \protected@edef\@glo@text{\csname glo@#1@first\endcsname}%
2622 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Get the plural form (as specified by the firstplural key when the entry was defined).

`\glsentryfirstplural`

```
2623 \newcommand*{\glsentryfirstplural}[1]{%
2624 \csname glo@#1@firstpl\endcsname}
```

`\Glsentryfirstplural`

```
2625 \newcommand*{\Glsentryfirstplural}[1]{%
2626 \protected@edef\@glo@text{\csname glo@#1@firstpl\endcsname}%
2627 \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`

```
2628 \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`

```
2629 \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.

```
2630 \newcommand*{\glsentryuseri}[1]{\csname glo@#1@useri\endcsname}
```

```

\Glsentryuseri
2631 \newcommand*{\Glsentryuseri}[1]{%
2632 \protected@edef\@glo@text{\csname glo@#1@useri\endcsname}%
2633 \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.
2634 \newcommand*{\glsentryuserii}[1]{\csname glo@#1@userii\endcsname}

\Glsentryuserii
2635 \newcommand*{\Glsentryuserii}[1]{%
2636 \protected@edef\@glo@text{\csname glo@#1@userii\endcsname}%
2637 \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.
2638 \newcommand*{\glsentryuseriii}[1]{\csname glo@#1@useriii\endcsname}

\Glsentryuseriii
2639 \newcommand*{\Glsentryuseriii}[1]{%
2640 \protected@edef\@glo@text{\csname glo@#1@useriii\endcsname}%
2641 \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.
2642 \newcommand*{\glsentryuseriv}[1]{\csname glo@#1@useriv\endcsname}

\Glsentryuseriv
2643 \newcommand*{\Glsentryuseriv}[1]{%
2644 \protected@edef\@glo@text{\csname glo@#1@useriv\endcsname}%
2645 \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.
2646 \newcommand*{\glsentryuserv}[1]{\csname glo@#1@userv\endcsname}

\Glsentryuserv
2647 \newcommand*{\Glsentryuserv}[1]{%
2648 \protected@edef\@glo@text{\csname glo@#1@userv\endcsname}%
2649 \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.
2650 \newcommand*{\glsentryuservi}[1]{\csname glo@#1@uservi\endcsname}

\Glsentryuservi
2651 \newcommand*{\Glsentryuservi}[1]{%
2652 \protected@edef\@glo@text{\csname glo@#1@uservi\endcsname}%
2653 \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.

```
2654 \newcommand*\glsentryshort}[1]{\csname glo@#1@short\endcsname}
```

`\Glsentryshort`

```
2655 \newcommand*\Glsentryshort}[1]{%
2656 \protected@edef\@glo@text{\csname glo@#1@short\endcsname}%
2657 \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.

```
2658 \newcommand*\glsentryshortpl}[1]{\csname glo@#1@shortpl\endcsname}
```

`\Glsentryshortpl`

```
2659 \newcommand*\Glsentryshortpl}[1]{%
2660 \protected@edef\@glo@text{\csname glo@#1@shortpl\endcsname}%
2661 \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.

```
2662 \newcommand*\glsentrylong}[1]{\csname glo@#1@long\endcsname}
```

`\Glsentrylong`

```
2663 \newcommand*\Glsentrylong}[1]{%
2664 \protected@edef\@glo@text{\csname glo@#1@long\endcsname}%
2665 \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.

```
2666 \newcommand*\glsentrylongpl}[1]{\csname glo@#1@longpl\endcsname}
```

`\Glsentrylongpl`

```
2667 \newcommand*\Glsentrylongpl}[1]{%
2668 \protected@edef\@glo@text{\csname glo@#1@longpl\endcsname}%
2669 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Short cut macros to access full form:

`\glsentryfull`

```
2670 \newcommand*\glsentryfull}[1]{%
2671 \glsentrylong{#1}\space(\glsentryshort{#1})%
2672 }
```

`\Glsentryfull`

```
2673 \newcommand*\Glsentryfull}[1]{%
2674 \Glsentrylong{#1}\space(\glsentryshortpl{#1})%
2675 }
```

```

\glsentryfullpl
2676 \newcommand*{\glsentryfullpl}[1]{%
2677   \glsentrylongpl{#1}\space(\glsentryshort{#1})%
2678 }

\Glsentryfullpl
2679 \newcommand*{\Glsentryfullpl}[1]{%
2680   \Glsentrylongpl{#1}\space(\glsentryshortpl{#1})%
2681 }

```

`\gls hyperlink` Provide a hyperlink to a glossary entry without adding information to the glossary file. The entry needs to be added using a command like `\gls link` or `\gls add` 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.

```

2682 \newcommand*{\gls hyperlink}[2][\glsentrytext{\@glo@label}]{%
2683   \def\@glo@label{#2}%
2684   \@gls link{glo:#2}{#1}}

```

1.11 Adding an entry to the glossary without generating text

The following keys are provided for `\gls add` and `\gls addall`:

```

2685 \define@key{glossadd}{counter}{\def\@gls@counter{#1}}
2686 \define@key{glossadd}{format}{\def\@glsnumberformat{#1}}

```

This key is only used by `\gls addall`:

```

2687 \define@key{glossadd}{types}{\def\@glo@type{#1}}

```

`\gls add[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
2688 \newrobustcmd*{\glsadd}[2][ ]{%
2689   \glsdoifexists{#2}%
2690   {%
2691     \def\@glsnumberformat{glsnumberformat}%
2692     \edef\@gls@counter{\csname glo@#2@counter\endcsname}%
2693     \setkeys{glossadd}{#1}%
2694     Store the entry's counter in \theglsentrycounter
2695     \@gls@saveentrycounter
2696     \@do@wrglossary{#2}%
2697   }%

```

`\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`

```
2698 \newrobustcmd*{\glsaddall}[1] [] {%
2699 \edef\@glo@type{\@glo@types}%
2700 \setkeys{glossadd}{#1}%
2701 \forallglsentries[\@glo@type]{\@glo@entry}{%
2702 \glsadd[#1]{\@glo@entry}}%
2703 }
```

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.

```
2704 \edef\glsopenbrace{\expandafter\@gobble\string\{}
```

`\glsclousebrace` Define `\glsclousebrace` to make it easier to write an opening brace to a file.

```
2705 \edef\glsclousebrace{\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.

```
2706 \edef\glsquote#1{\string"#1\string"}
```

`\@glsfirstletter` Define the first letter to come after the digits 0, ..., 9. Only required for `xindy`.

```
2707 \ifglsxindy
2708 \newcommand*{\@glsfirstletter}{A}
2709 \fi
```

```

\GlsSetXdyFirstLetterAfterDigits Sets the first letter to come after the digits 0,...,9.
2710 \ifglxsindy
2711 \newcommand*\GlsSetXdyFirstLetterAfterDigits}[1]{%
2712 \renewcommand*\@glsglsfirstletter}{#1}}
2713 \else
2714 \newcommand*\GlsSetXdyFirstLetterAfterDigits}[1]{%
2715 \glsglnoindywarning\GlsSetXdyFirstLetterAfterDigits}
2716 \fi

\@glsglsminrange Define the minimum number of successive location references to merge into a
range.
2717 \newcommand*\@glsglsminrange}{2}

\GlsSetXdyMinRangeLength 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.
2718 \ifglxsindy
2719 \newcommand*\GlsSetXdyMinRangeLength}[1]{%
2720 \renewcommand*\@glsglsminrange}{#1}}
2721 \else
2722 \newcommand*\GlsSetXdyMinRangeLength}[1]{%
2723 \glsglnoindywarning\GlsSetXdyMinRangeLength}
2724 \fi

\writeist
2725 \ifglxsindy
Code to use if xindy is required.
2726 \def\writeist{%
Update attributes list
2727 \@glsgls@addpredefinedattributes
Open the file.
2728 \openout\glswrite=\istfilename
Write header comment at the start of the file
2729 \write\glswrite{;; xindy style file created by the glossaries
2730 package}%
2731 \write\glswrite{;; for document '\jobname' on
2732 \the\year-\the\month-\the\day}%
Specify the required styles
2733 \write\glswrite{^^J; required styles^^J}
2734 \@for\@xdystyle:=\@xdyrequiredstyles\do{%
2735 \ifx\@xdystyle\@empty
2736 \else
2737 \protected@write\glswrite}{(require
2738 \string"\@xdystyle.xdy\string")}%
2739 \fi
2740 }%

```

List the allowed attributes (possible values used by the format key)

```
2741 \write\glswrite{^^J%
2742 ; list of allowed attributes (number formats)^^J}%
2743 \write\glswrite{(define-attributes ((\xdyattributes)))}%
```

Define any additional alphabets

```
2744 \write\glswrite{^^J; user defined alphabets^^J}%
2745 \write\glswrite{\@xdyuseralphabets}%
```

Define location classes.

```
2746 \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.

```
2747 \@for\@gls@classI:=\@gls@xdy@locationlist\do{%
```

Case were $\langle Hprefix \rangle$ is empty:

```
2748 \protected@write\glswrite{}{(define-location-class
2749 \string"\@gls@classI\string"^^J\space\space\space
2750 (
2751 :sep "{{"
2752 \csname @gls@xdy@Lclass@\@gls@classI\endcsname\space
2753 :sep "}"
2754 )
2755 ^^J\space\space\space
2756 :min-range-length \@glsminrange^^J%
2757 )
2758 }%
```

Nested iteration over all classes:

```
2759 {%
2760 \@for\@gls@classII:=\@gls@xdy@locationlist\do{%
2761 \protected@write\glswrite{}{(define-location-class
2762 \string"\@gls@classII-\@gls@classI\string"
2763 ^^J\space\space\space
2764 (
2765 :sep "{"
2766 \csname @gls@xdy@Lclass@\@gls@classII\endcsname\space
2767 :sep "}{{"
2768 \csname @gls@xdy@Lclass@\@gls@classI\endcsname\space
2769 :sep "}"
2770 )
2771 ^^J\space\space\space
2772 :min-range-length \@glsminrange^^J%
2773 )
2774 }%
2775 }%
2776 }%
2777 }%
```

User defined location classes (needs checking for new location format).

```
2778 \write\glswrite{^^J; user defined location classes}%
```

```

2779     \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.)
2780     \write\glswrite{^^J; define cross-reference class^^J}%
2781     \write\glswrite{(define-crossref-class \string"see\string"
2782         :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.)
2783     \write\glswrite{(markup-crossref-list
2784         :class \string"see\string"^^J\space\space\space
2785         :open \string"\string\glsseeformat\string"
2786         :close \string"{}\string")}%
List the order to sort the classes.
2787     \write\glswrite{^^J; define the order of the location classes}%
2788     \write\glswrite{(define-location-class-order
2789         (\@xdylocationclassorder))}%
Specify what to write to the start and end of the glossary file.
2790     \write\glswrite{^^J; define the glossary markup^^J}%
2791     \write\glswrite{(markup-index^^J\space\space\space
2792         :open \string"\string
2793         \glossarysection[\string\glossarytoctitle]{\string
2794         \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)
2795     \@for\@this@ctr:=\@xdycounters\do{%
2796     {%
2797         \@for\@this@attr:=\@xdyattributelist\do{%
2798             \protected@write\glswrite{ }\string\providecommand*%
2799             \expandafter\string
2800             \csname glsX\@this@ctr X\@this@attr\endcsname[2]%
2801             {%
2802                 \string\setentrycounter
2803                 [\expandafter@gobble\string\#1]{\@this@ctr}%
2804                 \expandafter\string
2805                 \csname\@this@attr\endcsname
2806                 {\expandafter@gobble\string\#2}%
2807             }%
2808         }%
2809     }%
2810 }%
2811 }%

```

Add the end part of the open tag and the rest of the markup-index information:

```
2812 \write\glswrite{%
2813   \string\begin
2814   {theglossary}\string\glossaryheader\string~n\string" ^^J\space
2815   \space\space:close \string"\expandafter@gobble
2816   \string%\string~n\string
2817   \end{theglossary}\string\glossarypostamble
2818   \string~n\string" ^^J\space\space\space
2819   :tree)}%
```

Specify what to put between letter groups

```
2820 \write\glswrite{(markup-letter-group-list
2821   :sep \string"\string\glsgroupskip\string~n\string")}%
```

Specify what to put between entries

```
2822 \write\glswrite{(markup-indexentry
2823   :open \string"\string\relax \string\glsresetentrylist
2824   \string~n\string")}%
```

Specify how to format entries

```
2825 \write\glswrite{(markup-locclass-list :open
2826   \string"\glsopenbrace\string\glossaryentrynumbers
2827   \glsopenbrace\string\relax\space \string"^^J\space\space\space
2828   :sep \string", \string"
2829   :close \string"\glsclosebrace\glsclosebrace\string")}%
```

Specify how to separate location numbers

```
2830 \write\glswrite{(markup-locref-list
2831   :sep \string"\string\delimN\space\string")}%
```

Specify how to indicate location ranges

```
2832 \write\glswrite{(markup-range
2833   :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.

```
2834 \@onelevel@sanitize\gls@suffixF
2835 \@onelevel@sanitize\gls@suffixFF
2836 \ifx\gls@suffixF\@empty
2837 \else
2838   \write\glswrite{(markup-range
2839     :close "\gls@suffixF" :length 1 :ignore-end)}%
2840 \fi
2841 \ifx\gls@suffixFF\@empty
2842 \else
2843   \write\glswrite{(markup-range
2844     :close "\gls@suffixFF" :length 2 :ignore-end)}%
2845 \fi
```

Specify how to format locations.

```
2846 \write\glswrite{^^J; define format to use for locations^^J}%
2847 \write\glswrite{\@xdylocref}%
```

Specify how to separate letter groups.

```
2848 \write\glswrite{^^J; define letter group list format^^J}%
2849 \write\glswrite{(markup-letter-group-list
2850 :sep \string"\string\glsgroupskip\string~n\string")}%
```

Define letter group headings.

```
2851 \write\glswrite{^^J; letter group headings^^J}%
2852 \write\glswrite{(markup-letter-group
2853 :open-head \string"\string\glsgroupheading
2854 \glsopenbrace\string"^^J\space\space\space
2855 :close-head \string"\glsfclosebrace\string")}%
```

Define additional letter groups.

```
2856 \write\glswrite{^^J; additional letter groups^^J}%
2857 \write\glswrite{\@xdylettergroups}%
```

Define additional sort rules

```
2858 \write\glswrite{^^J; additional sort rules^^J}
2859 \write\glswrite{\@xdysortrules}%
```

Close the style file

```
2860 \closeout\glswrite
```

Suppress any further calls.

```
2861 \let\writeist\relax
2862 }
2863 \else
```

Code to use if makeindex is required.

```
2864 \edef\@gls@actualchar{\string?}
2865 \edef\@gls@encapchar{\string|}
2866 \edef\@gls@levelchar{\string!}
2867 \edef\@gls@quotechar{\string"}
2868 \def\writeist{\relax
2869 \openout\glswrite=\istfilename
2870 \write\glswrite{\expandafter\@gobble\string}% makeindex style file
2871 created by the glossaries package}
2872 \write\glswrite{\expandafter\@gobble\string}% for document
2873 '\jobname' on \the\year-\the\month-\the\day}
2874 \write\glswrite{actual '\@gls@actualchar'}
2875 \write\glswrite{encap '\@gls@encapchar'}
2876 \write\glswrite{level '\@gls@levelchar'}
2877 \write\glswrite{quote '\@gls@quotechar'}
2878 \write\glswrite{keyword \string"\string\glossaryentry\string"}
2879 \write\glswrite{preamble \string"\string\glossarysection[\string
2880 \glossarytoctitle]{\string\glossarytitle}\string
2881 \glossarypreamble\string\n\string\begin{theglossary}\string
2882 \glossaryheader\string\n\string"}
2883 \write\glswrite{postamble \string"\string%\string\n\string
2884 \end{theglossary}\string\glossarypostamble\string\n
2885 \string"}
2886 \write\glswrite{group_skip \string"\string\glsgroupskip\string\n
```

```

2887     \string"}
2888 \write\glswrite{item_0 \string"\string%\string\n\string"}
2889 \write\glswrite{item_1 \string"\string%\string\n\string"}
2890 \write\glswrite{item_2 \string"\string%\string\n\string"}
2891 \write\glswrite{item_01 \string"\string%\string\n\string"}
2892 \write\glswrite{item_x1
2893     \string"\string\relax \string\glsresetentrylist\string\n
2894     \string"}
2895 \write\glswrite{item_12 \string"\string%\string\n\string"}
2896 \write\glswrite{item_x2
2897     \string"\string\relax \string\glsresetentrylist\string\n
2898     \string"}

2899 \write\glswrite{delim_0 \string"\string\{\string
2900     \glossaryentrynumbers\string\{\string\relax \string"}
2901 \write\glswrite{delim_1 \string"\string\{\string
2902     \glossaryentrynumbers\string\{\string\relax \string"}
2903 \write\glswrite{delim_2 \string"\string\{\string
2904     \glossaryentrynumbers\string\{\string\relax \string"}
2905 \write\glswrite{delim_t \string"\string\}\string}\string"}
2906 \write\glswrite{delim_n \string"\string\delimN \string"}
2907 \write\glswrite{delim_r \string"\string\delimR \string"}
2908 \write\glswrite{headings_flag 1}
2909 \write\glswrite{heading_prefix
2910     \string"\string\glsgroupheading\string\{\string"}
2911 \write\glswrite{heading_suffix
2912     \string"\string}\string\relax
2913     \string\glsresetentrylist \string"}
2914 \write\glswrite{symhead_positive \string"glssymbols\string"}
2915 \write\glswrite{numhead_positive \string"glnumbers\string"}
2916 \write\glswrite{page_compositor \string"glscpositor\string"}
2917 \@gls@escbsdq\gls@suffixF
2918 \@gls@escbsdq\gls@suffixFF
2919 \ifx\gls@suffixF\@empty
2920 \else
2921     \write\glswrite{suffix_2p \string"\gls@suffixF\string"}
2922 \fi
2923 \ifx\gls@suffixFF\@empty
2924 \else
2925     \write\glswrite{suffix_3p \string"\gls@suffixFF\string"}
2926 \fi
2927 \closeout\glswrite
2928 \let\writeist\relax
2929 }
2930 \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
2931 \newcommand{\noist}{%

```

Update attributes list

```
2932 \gls@addpredefinedattributes
2933 \let\writeist\relax
2934 }
```

`\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`

```
2935 \newcommand*\@makeglossary}[1]{%
2936 \ifglossaryexists{#1}%
2937 {%
    Only create a new write if savewrites=false otherwise create a token to collect the
    information.
2938     \ifglssavewrites
2939     \expandafter\newtoks\csname glo@#1@filetok\endcsname
2940     \else
2941     \expandafter\newwrite\csname glo@#1@file\endcsname
2942     \expandafter\@glsopenfile\csname glo@#1@file\endcsname{#1}%
2943     \fi
2944     \gls@renewglossary
2945     \writeist
2946 }%
2947 {%
2948 \PackageError{glossaries}%
2949 {Glossary type ‘#1’ not defined}%
2950 {New glossaries must be defined before using \string\makeglossary}%
2951 }%
2952 }
```

`\@glsopenfile` Open write file associated with the given glossary.

```
2953 \newcommand*\@glsopenfile}[2]{%
2954 \immediate\openout#1=\jobname.\csname @glotype@#2@out\endcsname
2955 \PackageInfo{glossaries}{Writing glossary file
2956 \jobname.\csname @glotype@#2@out\endcsname}%
2957 }
```

`\warn@nomakeglossaries` Issue warning that `\makeglossaries` hasn't been used.

```
2958 \newcommand*\warn@nomakeglossaries}{%
2959 \GlossariesWarningNoLine{\string\makeglossaries\space
2960 hasn't been used,^^Jthe glossaries will not be updated}%
2961 }
```

`\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`

```
2962 \newcommand*{\makeglossaries}{%
```

Write the name of the style file to the aux file (needed by `makeglossaries`)

```
2963 \protected@write\@auxout{}{\string\@istfilename{\istfilename}}%
```

```
2964 \protected@write\@auxout{}{\string\@glsorder{\glsorder}}%
```

Iterate through each glossary type and activate it.

```
2965 \@for\@glo@type:=\@glo@types\do{%
```

```
2966 \ifthenelse{\equal{\@glo@type}{}}{}{}%
```

```
2967 \@makeglossary{\@glo@type}}%
```

```
2968 }%
```

New glossaries must be created before `\makeglossaries` so disable `\newglossary`.

```
2969 \renewcommand*\newglossary[4] [] {}%
```

```
2970 \PackageError{glossaries}{New glossaries
```

```
2971 must be created before \string\makeglossaries}{You need
```

```
2972 to move \string\makeglossaries\space after all your
```

```
2973 \string\newglossary\space commands}}%
```

Any subsequent instances of this command should have no effect

```
2974 \let\@makeglossary\relax
```

```
2975 \let\makeglossary\relax
```

```
2976 \let\makeglossaries\relax
```

Disable all commands that have no effect after `\makeglossaries`

```
2977 \@disable@onlypremakeg
```

Suppress warning about no `\makeglossaries`

```
2978 \let\warn@nomakeglossaries\relax
```

```
2979 }
```

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`

```
2980 \let\makeglossary\makeglossaries
```

If `\makeglossaries` hasn't been used, issue a warning. Also issue a warning if neither `\printglossaries` nor `\printglossary` have been used.

```
2981 \AtEndDocument{%
```

```
2982 \warn@nomakeglossaries
```

```
2983 \warn@noprintglossary
```

```
2984 }
```

1.13 Writing information to associated files

`\glswrite` The write used for style file also used for all other output files if `savewrites=true`.

```
2985 \newwrite\glswrite
```

`\istfile` Deprecated.

```
2986 \def\istfile{\glswrite}
```

At the end of the document, the files should be created if `savewrites=true`.

```
2987 \AtEndDocument{%
```

```
2988   \glswritefiles
```

```
2989 }
```

`\glswritefiles` Only write the files if `savewrites=true`

```
2990 \ifglssavewrites
```

```
2991   \newcommand*{\glswritefiles}{%
```

Iterate through all the glossaries

```
2992     \forallglossaries{\@glo@type}{%
```

```
2993       \edef\gls@tmp{\expandafter\the\csname glo@\@glo@type
```

```
2994 @filetok\endcsname}%
```

```
2995       \ifx\gls@tmp\@empty
```

```
2996         \ifx\@glo@type\glsdefaulttype
```

```
2997           \GlossariesWarningNoLine{Glossary ‘\@glo@type’ has no
```

```
2998             entries.^^JRemember to use package option ‘nomain’ if
```

```
2999 you
```

```
3000             don’t want to^^Juse the main glossary}%
```

```
3001         \else
```

```
3002           \GlossariesWarningNoLine{Glossary ‘\@glo@type’ has no
```

```
3003             entries}%
```

```
3004         \fi
```

```
3005       \else
```

```
3006         \@glsopenfile{\glswrite}{\@glo@type}%
```

```
3007         \immediate\write\glswrite{%
```

```
3008           \expandafter\the\csname glo@\@glo@type
```

```
3009 @filetok\endcsname}%
```

```
3010         \immediate\closeout\glswrite
```

```
3011       \fi
```

```
3012     }%
```

```
3013   }
```

```
3014 \else
```

```
3015   \let\glswritefiles\relax
```

```
3016 \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`

```
3017 \renewcommand*\glossary}[1][\glsdefaulttype]{%
3018 \glossary[#1]}
```

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`

```
3019 \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`

```
3020 \newcommand\@gls@renewglossary}{%
3021 \gdef\@glossary[##1]{\@bsphack\begingroup\@wrglossary{##1}}%
3022 \let\@gls@renewglossary\@empty
3023 }
```

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`

```
3024 \renewcommand*\@wrglossary}[2]{%
3025 \ifglssavewrites
3026 \protected@edef\@gls@tmp{\the\csname glo@#1@filetok\endcsname#2}%
3027 \expandafter\global\expandafter\csname glo@#1@filetok\endcsname
3028 \expandafter{\@gls@tmp~J}%
3029 \else
3030 \expandafter\protected@write\csname glo@#1@file\endcsname{}{#2}%
3031 \fi
3032 \endgroup\@esphack
3033 }
```

`\@do@wrglossary` Write the glossary entry in the appropriate format. (Need to set `\@glsnumberformat` and `\@gls@counter` prior to use.) The argument is the entry's label.

```
3034 \newcommand\@do@wrglossary}[1]{%
```

Get the location and escape any special characters

```
3035 \protected@edef\@gls@locref{\theHglentrycounter}%
3036 \@gls@checkmkidxchars\@gls@locref
```

Check if the hyper-location is the same as the location and set the hyper prefix.

```
3037 \expandafter\ifx\theHglentrycounter\theHglentrycounter
3038 \def\@gls@counterprefix{}%
3039 \else
3040 \protected@edef\@gls@Hlocref{\theHglentrycounter}%
3041 \@gls@checkmkidxchars\@gls@Hlocref
3042 \edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix
```

```

3043     {\@glslocref}{\@glsHlocref}%
3044   }%
3045   \@do@gls@getcounterprefix
3046 \fi

Determine whether to use xindy or makeindex syntax
3047 \ifglsxindy

Need to determine if the formatting information starts with a ( or ) indicating a
range.
3048 \expandafter\@glo@check@mkidxrangechar\@glsnumberformat\@nil
3049 \def\@glo@range{}%
3050 \expandafter\if\@glo@prefix\relax
3051   \def\@glo@range{:open-range}%
3052 \else
3053   \expandafter\if\@glo@prefix\relax
3054   \def\@glo@range{:close-range}%
3055 \fi
3056 \fi

Write to the glossary file using xindy syntax.
3057 \glossary[\csname glo@#1@type\endcsname]{%
3058 (indexentry :tkey (\csname glo@#1@index\endcsname)
3059   :locref \string"{\@glo@counterprefix}{\@glslocref}\string" %
3060   :attr \string"\@gls@counter\@glo@suffix\string"
3061   \@glo@range
3062 )
3063 }%
3064 \else

Convert the format information into the format required for makeindex
3065 \@set@glo@numformat{\@glo@numfmt}{\@gls@counter}{\@glsnumberformat}%
3066   {\@glo@counterprefix}%

Write to the glossary file using makeindex syntax.
3067 \glossary[\csname glo@#1@type\endcsname]{%
3068 \string\glossaryentry{\csname glo@#1@index\endcsname
3069   \@gls@encapchar\@glo@numfmt}{\theglentrycounter}}%
3070 \fi
3071 }

```

`\@gls@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.)

```

3072 \newcommand*\@gls@getcounterprefix[2]{%
3073   \edef\@gls@thisloc{#1}\edef\@gls@thisHloc{#2}%
3074   \ifx\@gls@thisloc\@gls@thisHloc
3075     \def\@glo@counterprefix{}%
3076   \else

```

```

3077 \def\gls@get@counterprefix##1.#1##2\end@getprefix{%
3078 \def\glo@tmp{##2}%
3079 \ifx\glo@tmp\@empty
3080 \def\glo@counterprefix{%
3081 \else
3082 \def\glo@counterprefix{##1}%
3083 \fi
3084 }%
3085 \gls@get@counterprefix#2.#1\end@getprefix
3086 \fi
3087 }

```

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.

```

3088 \newcommand{\do@seeglossary}[2]{%
3089 \def\gls@xref{#2}%
3090 \@onelevel@sanitize\gls@xref
3091 \gls@checkmkidxchars\gls@xref
3092 \ifglsxindy
3093 \glossary[\csname glo@#1@type\endcsname]{%
3094 (indexentry
3095 :tkey (\csname glo@#1@index\endcsname)
3096 :xref (\string"\gls@xref\string")
3097 :attr \string"see\string"
3098 )
3099 }%
3100 \else
3101 \glossary[\csname glo@#1@type\endcsname]{%
3102 \string\glossaryentry{\csname glo@#1@index\endcsname
3103 \gls@encapchar glsseeformat\gls@xref}{Z}}%
3104 \fi
3105 }

```

`\gls@fixbraces` If no optional argument is specified, list needs to be enclosed in a set of braces.

```

3106 \def\gls@fixbraces#1#2#3\@nil{%
3107 \ifx#2[\relax
3108 \def#1{#2#3}%
3109 \else
3110 \def#1{{#2#3}}%
3111 \fi
3112 }

```

```

\glssee \glssee{\label}{\langle cross-ref list \rangle}
3113 \newcommand*\glssee[3][\seename]{%
3114 \do@seeglossary{#2}{[#1]{#3}}
3115 \newcommand*\glssee[3][\seename]{%

```

```

3116 \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.
3117 \newcommand*{\glsseeformat}[3][\seename]{\emph{#1} \glsseelist{#2}}

\glsseelist \glsseelist{<list>} formats list of entry labels.
3118 \newcommand*{\glsseelist}[1]{%
      If there is only one item in the list, set the last separator to do nothing.
3119 \let\@gls@dolast\relax
      Don’t display separator on the first iteration of the loop
3120 \let\@gls@donext\relax
      Iterate through the labels
3121 \@for\@gls@thislabel:=#1\do{%
      Check if on last iteration of loop
3122 \ifx\@xfor@nextelement\@nnil
3123 \@gls@dolast
3124 \else
3125 \@gls@donext
3126 \fi
      display the entry for this label
3127 \glsseeitem{\@gls@thislabel}%
      Update separators
3128 \let\@gls@dolast\glsseelastsep
3129 \let\@gls@donext\glsseesep
3130 }%
3131 }

\glsseelastsep Separator to use between penultimate and ultimate entries in a cross-referencing
                list.
3132 \newcommand*{\glsseelastsep}{\space\andname\space}

\glsseesep Separator to use between entires in a cross-referencing list.
3133 \newcommand*{\glsseesep}{, }

\glsseeitem \glsseeitem{<label>} formats individual entry in a cross-referencing list.
3134 \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.)
3135 \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.

`\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.)

```
3136 \def\warn@noprintglossary{\GlossariesWarningNoLine{No
3137 \string\printglossary\space or \string\printglossaries\space
3138 found.^^JThis document will not have a glossary}}
```

`\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.

```
3139 \ifcsundef{printglossary}{}%
3140 {%
```

If `\printglossary` is already defined, issue a warning and undefine it.

```
3141 \GlossariesWarning{Overriding \string\printglossary}%
3142 \undef\printglossary
3143 }
```

`\printglossary` has an optional argument. The default value is to set the glossary type to the main glossary.

```
3144 \newcommand*{\printglossary}[1][type=\glsdefaulttype]{%
```

If `xindy` is being used, need to find the root language for `makeglossaries` to pass to `xindy`.

```
3145 \ifglxindy\findrootlanguage\fi
```

Set up defaults.

```
3146 \def@glo@type{\glsdefaulttype}%
3147 \def@glossarytitle{\csname @glo@type @title\endcsname}%
3148 \let@org@glossarytitle@glossarytitle
3149 \def@glossarystyle{}%
3150 \def@gls@dotoc@title{\glssettoctitle{@glo@type}}%
```

Store current value of `\glossaryentrynumbers`. (This may be changed via the optional argument)

```
3151 \let@org@glossaryentrynumbers@glossaryentrynumbers
```

Localise the effects of the optional argument

```
3152 \bgroup
```

Determine settings specified in the optional argument.

```
3153 \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)

```
3154 \ifx\glossarytitle\org@glossarytitle
3155 \else
3156   \expandafter\let\csname @glo@type@\@glo@type @title\endcsname
3157     \glossarytitle
3158 \fi
```

Allow a high-level user command to indicate the current glossary

```
3159 \let\currentglossary\@glo@type
```

Enable individual number lists to be suppressed.

```
3160 \let\org@glossaryentrynumbers\glossaryentrynumbers
3161 \let\glsnonextpages\glsnonextpages
```

Enable individual number list to be activated:

```
3162 \let\glsnextpages\glsnextpages
```

Enable suppression of description terminators.

```
3163 \let\nopostdesc\@nopostdesc
```

Set up the entry for the TOC

```
3164 \gls@dotoc@title
```

Set the glossary style

```
3165 \@glossarystyle
```

Some macros may end up being expanded into internals in the glossary, so need to make @ a letter.

```
3166 \makeatletter
```

Input the glossary file, if it exists.

```
3167 \@input@{\jobname.\csname @glo@type@\@glo@type @in\endcsname}%
```

If the glossary file doesn't exist, do \null. (This ensures that the page is shipped out and all write commands are done.) This might produce an empty page, but at this point the document isn't complete, so it shouldn't matter.

```
3168 \IfFileExists{\jobname.\csname @glo@type@\@glo@type @in\endcsname}{}%
3169 {\null}%
```

If xindy is being used, need to write the language dependent information to the .aux file for makeglossaries.

```
3170 \ifglsxindy
3171   \ifcsundef{@xdy@\@glo@type @language}%
3172   {%
3173     \protected@write\@auxout{}{%
3174       \string\@xdy@language{\@glo@type}{\@xdy@main@language}}%
3175   }%
3176   {%
3177     \protected@write\@auxout{}{%
3178       \string\@xdy@language{\@glo@type}{\csname @xdy@\@glo@type
3179         @language\endcsname}}%
3180   }%
```

```

3181     \protected@write\@auxout{}{%
3182         \string\@gls@codepage{\@glo@type}{\gls@codepage}}%
3183     \fi
3184 \egroup
    Reset \glossaryentrynumbers
3185 \global\let\glossaryentrynumbers\@org@glossaryentrynumbers
    Suppress warning about no \printglossary
3186 \global\let\warn@noprintglossary\relax
3187 }

```

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`

```

3188 \newcommand*{\printglossaries}{%
3189 \foralllglossaries{\@glo@type}{\printglossary[type=\@glo@type]}}

```

The keys that can be used in the optional argument to `\printglossary` are as follows: The `type` key sets the glossary type.

```

3190 \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`.

```

3191 \define@key{printgloss}{title}{\def\glossarytitle{#1}}

```

The `toctitle` sets the text used for the relevant entry in the table of contents.

```

3192 \define@key{printgloss}{toctitle}{\def\glossarytoctitle{#1}}%
3193 \let\gls@dotoc@title\relax
3194 }

```

The `style` key sets the glossary style (but only for the given glossary).

```

3195 \define@key{printgloss}{style}{%
3196   \ifcsundef{glsstyle@#1}%
3197   {%
3198     \PackageError{glossaries}%
3199     {Glossary style ‘#1’ undefined}{%
3200     }%
3201   }%
3202   \def\@glossarystyle{\csname @glsstyle@#1\endcsname}%
3203 }%
3204 }

```

The `numberedsection` key determines if this glossary should be in a numbered section.

```

3205 \define@choicekey{printgloss}{numberedsection}[\val\nr]{%
3206 false,nolabel,autolabel}[nolabel]{%
3207 \ifcase\nr\relax
3208 \renewcommand*{\@@glossarysecstar}{*}%
3209 \renewcommand*{\@@glossaryseclabel}{}%
3210 \or
3211 \renewcommand*{\@@glossarysecstar}{}%
3212 \renewcommand*{\@@glossaryseclabel}{}%
3213 \or
3214 \renewcommand*{\@@glossarysecstar}{*}%
3215 \renewcommand*{\@@glossaryseclabel}{\label{\glsautoprefix\@glo@type}}%
3216 \fi}

```

The `nonumberlist` key determines if this glossary should have a number list.

```

3217 \define@boolkey{printgloss}[gls]{nonumberlist}[true]{%
3218 \ifglsnonumberlist
3219 \def\glossaryentrynumbers##1{%
3220 \else
3221 \def\glossaryentrynumbers##1{##1}%
3222 \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.

```

3223 \newcommand*{\@glsnonextpages}{%
3224 \gdef\glossaryentrynumbers##1{%
3225 \glsresetentrylist}}

```

`\@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.

```

3226 \newcommand*{\@glsnextpages}{%
3227 \gdef\glossaryentrynumbers##1{%
3228 ##1\glsresetentrylist}}

```

`\glsresetentrylist` Resets `\glossaryentrynumbers`

```

3229 \newcommand*{\glsresetentrylist}{%
3230 \global\let\glossaryentrynumbers\org@glossaryentrynumbers}

```

`\glsnonextpages` Outside of `\printglossary` this does nothing.

```

3231 \newcommand*{\glsnonextpages}{}

```

`\glsnextpages` Outside of `\printglossary` this does nothing.

```
3232 \newcommand*\glsnextpages{}
```

`glossaryentry` If the `entrycounter` package option has been used, define a counter to number each level 0 entry.

```
3233 \ifglentrycounter
3234   \ifx\@gls@counterwithin\@empty
3235     \newcounter{glossaryentry}
3236   \else
3237     \newcounter{glossaryentry}[\@gls@counterwithin]
3238   \fi
3239   \def\theHglossaryentry{\currentglossary.\theglossaryentry}
3240 \fi
```

`glossarysubentry` If the `subentrycounter` package option has been used, define a counter to number each level 1 entry.

```
3241 \ifglssubentrycounter
3242   \ifglentrycounter
3243     \newcounter{glossarysubentry}[glossaryentry]
3244   \else
3245     \newcounter{glossarysubentry}
3246   \fi
3247   \def\theHglossarysubentry{\currentglssubentry.\theglossarysubentry}
3248 \fi
```

`\glsresetsubentrycounter` Resets the `glossarysubentry` counter.

```
3249 \ifglssubentrycounter
3250   \newcommand*\glsresetsubentrycounter}{%
3251     \setcounter{glossarysubentry}{0}%
3252   }
3253 \else
3254   \newcommand*\glsresetsubentrycounter{}
3255 \fi
```

`\glsstepentry` Advance the `glossaryentry` counter if in use. The argument is the label associated with the entry.

```
3256 \ifglentrycounter
3257   \newcommand*\glsstepentry}[1]{%
3258     \refstepcounter{glossaryentry}%
3259     \label{glsentry-#1}%
3260   }
3261 \else
3262   \newcommand*\glsstepentry}[1]{}
3263 \fi
```

`\glsstepsubentry` Advance the `glossarysubentry` counter if in use. The argument is the label associated with the subentry.

```
3264 \ifglssubentrycounter
3265   \newcommand*\glsstepsubentry}[1]{%
```

```

3266     \def\currentglssubentry{#1}%
3267     \refstepcounter{glossarysubentry}%
3268     \label{glentry-#1}%
3269   }
3270 \else
3271   \newcommand*\glssubentry[1]{}
3272 \fi

```

`\glssrefentry` Reference the entry or sub-entry counter if in use, otherwise just do `\gls`.

```

3273 \ifglssentrycounter
3274   \newcommand*\glssrefentry[1]{\ref{glentry-#1}}
3275 \else
3276   \ifglssubentrycounter
3277     \newcommand*\glssrefentry[1]{\ref{glentry-#1}}
3278   \else
3279     \newcommand*\glssrefentry[1]{\gls{#1}}
3280   \fi
3281 \fi

```

`\glssentrycounterlabel` Defines how to display the glossaryentry counter.

```

3282 \ifglssentrycounter
3283   \newcommand*\glssentrycounterlabel{\theglossaryentry.\space}
3284 \else
3285   \newcommand*\glssentrycounterlabel{}
3286 \fi

```

`\glsssubentrycounterlabel` Defines how to display the glossarysubentry counter.

```

3287 \ifglsssubentrycounter
3288   \newcommand*\glsssubentrycounterlabel{\theglossarysubentry}\space}
3289 \else
3290   \newcommand*\glsssubentrycounterlabel{}
3291 \fi

```

`\glssentryitem` Step and display glossaryentry counter, if appropriate.

```

3292 \ifglssentrycounter
3293   \newcommand*\glssentryitem[1]{%
3294     \glssstepentry{#1}\glssentrycounterlabel
3295   }
3296 \else
3297   \newcommand*\glssentryitem[1]{\glssresetsubentrycounter}
3298 \fi

```

`\glsssubentryitem` Step and display glossarysubentry counter, if appropriate.

```

3299 \ifglsssubentrycounter
3300   \newcommand*\glsssubentryitem[1]{%
3301     \glssstepsubentry{#1}\glsssubentrycounterlabel
3302   }
3303 \else
3304   \newcommand*\glsssubentryitem[1]{}
3305 \fi

```

`theglossary` If the `theglossary` environment has already been defined, a warning will be issued. This environment should be redefined by glossary styles.

```
3306 \ifcsundef{theglossary}%
3307 {%
3308   \newenvironment{theglossary}{}{}%
3309 }%
3310 {%
3311   \GlossariesWarning{overriding ‘theglossary’ environment}%
3312   \renewenvironment{theglossary}{}{}%
3313 }
```

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`

```
3314 \newcommand*{\glossaryheader}{}
```

`\glstarget` `\glstarget{<label>}{<name>}`

Provide user interface to `\@glstarget` to make it easier to modify the glossary style in the document.

```
3315 \newcommand*{\glstarget}[2]{\@glstarget{glo:#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>`.

```
3316 \newcommand*{\glossaryentryfield}[5]{%
3317 \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.)

```
3318 \newcommand*{\glossarysubentryfield}[6]{%
3319 \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`

```
3320 \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: `glsymbols`, `glsnumbers`, `A`, ..., `Z`. Glossary styles must redefine this command. (In between groups, `\glsgroupheading` comes immediately after `\glsgroupskip`.)

`\glsgroupheading`

```
3321 \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 `glsymbols` produces `\glsymbolsgroupname`, the group labelled `glsnumbers` produces `\glsnumbersgroupname` and all the other groups simply produce their label. As mentioned above, the group labels are: `glsymbols`, `glsnumbers`, `A`, ..., `Z`. If you want to redefine the group titles, you will need to redefine this command.

`\glsgetgrouptitle`

```
3322 \newcommand*\glsgetgrouptitle}[1]{%
3323 \ifcsundef{#1groupname}{#1}{\csname #1groupname\endcsname}%
3324 }
```

```
\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`

```
3325 \newcommand*\glsgetgrouplabel}[1]{%
```

```
3326 \ifthenelse{\equals{#1}{\glssymbolsgroupname}}{\glssymbols}{%
3327 \ifthenelse{\equals{#1}{\glsnumbersgroupname}}{\glsnumbers}{#1}}
```

The command `\setentrycounter` sets the entry's associated counter (required by `\glsnumber` etc.) `\glslink` and `\glsadd` encode the `\glossary` argument so that the relevant counter is set prior to the formatting command.

`\setentrycounter`

```
3328 \newcommand*\setentrycounter}[2] []{%
3329   \def\@glo@counterprefix{#1}%
3330   \ifx\@glo@counterprefix\empty
3331     \def\@glo@counterprefix{.}%
3332   \else
3333     \def\@glo@counterprefix{.#1.}%
3334   \fi
3335   \def\glsentrycounter{#2}%
3336 }
```

The current glossary style can be set using `\glossarystyle{<style>}`.

`\glossarystyle`

```
3337 \newcommand*\glossarystyle}[1]{%
3338   \ifcsundef{@glsstyle@#1}%
3339   {%
3340     \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
3341   }%
3342   {%
3343     \csname @glsstyle@#1\endcsname
3344   }%
3345 }
```

`\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.

```
3346 \newcommand*\newglossarystyle}[2]{%
3347   \ifcsundef{@glsstyle@#1}%
3348   {%
3349     \expandafter\def\csname @glsstyle@#1\endcsname{#2}%
3350   }%
3351   {%
3352     \PackageError{glossaries}{Glossary style ‘#1’ is already defined}{}%
3353   }%
3354 }
```

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`

```
3355 \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`

```
3356 \ifcsundef{hyperlink}%
3357 {%
3358   \def\glshypernumber#1{#1}%
3359 }%
3360 {%
3361   \def\glshypernumber#1{\@glshypernumber#1\nohyperpage{}}\@nil}
3362 }
```

`\@glshypernumber` This code was provided by Heiko Oberdiek to allow material to be attached to the location.

```
3363 \def\@glshypernumber#1\nohyperpage#2#3\@nil{%
3364   \ifx\#1\%
3365     \else
3366       \@delimR#1\delimR\delimR\%
3367     \fi
3368   \ifx\#2\%
3369     \else
3370       #2%
3371     \fi
3372   \ifx\#3\%
3373     \else
3374       \@glshypernumber#3\@nil
3375     \fi
3376 }
```

`\@delimR` displays a range of numbers for the counter whose name is given by `\@gls@counter` (which must be set prior to using `\gls@hypernumber`).

`\@delimR`

```
3377 \def\@delimR#1\delimR #2\delimR #3\{\%
3378 \ifx\#2\%
3379 \@delimN{#1}%
3380 \else
3381 \@gls@numberlink{#1}\delimR\@gls@numberlink{#2}%
3382 \fi}
```

`\@delimN` displays a list of individual numbers, instead of a range:

`\@delimN`

```
3383 \def\@delimN#1{\@delimN#1\delimN \delimN\}
3384 \def\@delimN#1\delimN #2\delimN#3\{\%
3385 \ifx\#3\%
3386 \@gls@numberlink{#1}%
3387 \else
3388 \@gls@numberlink{#1}\delimN\@gls@numberlink{#2}%
3389 \fi
3390 }
```

The following code is modified from `hyperref's \HyInd@pagelink` where the name of the counter being used is given by `\@gls@counter`.

```
3391 \def\@gls@numberlink#1{%
3392 \begingroup
3393 \toks@={}%
3394 \@gls@removespaces#1 \@nil
3395 \endgroup}
3396 \def\@gls@removespaces#1 #2\@nil{%
3397 \toks@=\expandafter{\the\toks@#1}%
3398 \ifx\#2\%
3399 \edef\x{\the\toks@}%
3400 \ifx\x\empty
3401 \else
3402 \hyperlink{\gls@entrycounter\@glo@counterprefix\the\toks@}%
3403 {\the\toks@}%
3404 \fi
3405 \else
3406 \@gls@ReturnAfterFi{%
3407 \@gls@removespaces#2\@nil
3408 }%
3409 \fi
3410 }
3411 \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
3412 \newcommand*\hyperrm[1]{\textrm{\glshypernumber{#1}}}

\hypersf
3413 \newcommand*\hypersf[1]{\textsf{\glshypernumber{#1}}}

\hypertt
3414 \newcommand*\hypertt[1]{\texttt{\glshypernumber{#1}}}

\hyperbf
3415 \newcommand*\hyperbf[1]{\textbf{\glshypernumber{#1}}}

\hypermd
3416 \newcommand*\hypermd[1]{\textmd{\glshypernumber{#1}}}

\hyperit
3417 \newcommand*\hyperit[1]{\textit{\glshypernumber{#1}}}

\hypersl
3418 \newcommand*\hypersl[1]{\textsl{\glshypernumber{#1}}}

\hyperup
3419 \newcommand*\hyperup[1]{\textup{\glshypernumber{#1}}}

\hypersc
3420 \newcommand*\hypersc[1]{\textsc{\glshypernumber{#1}}}

\hyperemph
3421 \newcommand*\hyperemph[1]{\emph{\glshypernumber{#1}}}

```

1.16 Acronyms

If the acronym package option is used, a new glossary called `acronym` is created

```

3422 \ifglsacronym
3423 \newglossary[alg]{acronym}{acr}{acn}{\acronymname}
    and \acronymtype is set to the name of this new glossary.
3424 \renewcommand*\acronymtype{acronym}
3425 \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.

```

3426 \newcommand{\oldacronym}[4][\gls@label]{%
3427   \def\gls@label{#2}%
3428   \newacronym[#4]{#1}{#2}{#3}%
3429   \ifcsundef{xspace}%
3430     {%
3431       \expandafter\edef\csname#1\endcsname{%
3432         \noexpand\@ifstar{\noexpand\Gls{#1}}{\noexpand\gls{#1}}}%
3433       }%
3434     }%
3435     {%
3436       \expandafter\edef\csname#1\endcsname{%
3437         \noexpand\@ifstar{\noexpand\Gls{#1}\noexpand\xspace}{%
3438           \noexpand\gls{#1}\noexpand\xspace}%
3439         }%
3440       }%
3441 }

```

`\newacronym[<key-val list>]{<label>}{<abbrev>}{<long>}`

This is a quick way of defining acronyms, all it does is call `\newglossaryentry` with the appropriate values. It sets the glossary type to `\acronymtype` which will be `acronym` if the package option `acronym` has been used, otherwise it will be the default glossary. Since `\newacronym` merely calls `\newglossaryentry`, the acronym is treated like any other glossary entry.

If you prefer a different format, you can redefine `\newacronym` as required. The optional argument can be used to override any of the settings.

This is just a stub. It's redefined by commands like `\SetDefaultAcronymStyle`.

`\newacronym`

```

3442 \newcommand{\newacronym}[4][[]]{

```

Set up some convenient short cuts. These need to be changed if `\newacronym` is changed (or if the description key is changed).

`\acrpluralsuffix`

Plural suffix used by `\newacronym`. This just defaults to `\glspluralsuffix` but is changed to include `\textup` if the `smallcaps` option is used, so that the suffix doesn't appear in small caps as it doesn't look right. For example, `ABCS` looks as though the "s" is part of the acronym, but `ABCs` looks as though the "s" is a plural

suffix. Since the entire text `abcs` is set in `\textsc`, `\textup` is need to cancel it out.

```
3443 \newcommand*\acrpluralsuffix{\glspluralsuffix}
```

The following are defined for compatibility with version 2.07 and earlier.

`\glsshortkey`

```
3444 \newcommand*\glsshortkey{short}
```

`\glsshortpluralkey`

```
3445 \newcommand*\glsshortpluralkey{shortplural}
```

`\glslongkey`

```
3446 \newcommand*\glslongkey{long}
```

`\glslongpluralkey`

```
3447 \newcommand*\glslongpluralkey{longplural}
```

`\acrfull` Full form of the acronym.

```
3448 \newrobustcmd*\acrfull{%
```

```
3449 \@ifstar\s@acrfull\@ns@acrfull
```

```
3450 }
```

```
3451 \newcommand*\s@acrfull[2] [] {%
```

```
3452 \new@ifnextchar[{\@acrfull{hyper=false,#1}{#2}}%
```

```
3453 \@acrfull{hyper=false,#1}{#2} []}%
```

```
3454 }
```

```
3455 \newcommand*\ns@acrfull[2] [] {%
```

```
3456 \new@ifnextchar[{\@acrfull{#1}{#2}}%
```

```
3457 \@acrfull{#1}{#2} []}%
```

```
3458 }
```

Low-level macro:

```
3459 \def\@acrfull#1#2[#3] {%
```

```
3460 \acrlinkfullformat{\@acrlong}{\@acrshort}{#1}{#2}{#3}%
```

```
3461 }
```

`\acrlinkfullformat` Format for full links like `\acrfull`. Syntax: `\acrlinkfullformat{<long cs>}{<short cs>}{<options>}{<label>}{<insert>}`

```
3462 \newcommand{\acrlinkfullformat}[5] {%
```

```
3463 \acrfullformat{#1}{#3}{#4}[#5]}{#2}{#3}{#4} []}%
```

```
3464 }
```

`\acrfullformat` Default full form is `<long>` (`<short>`).

```
3465 \newcommand{\acrfullformat}[2]{#1\space(#2)}
```

Default format for full acronym

`\Acrfull`

```
3466 \newrobustcmd*{\Acrfull}{%
3467 \@ifstar\s@Acrfull\ns@Acrfull
3468 }

3469 \newcommand*\s@Acrfull[2][]{%
3470 \new@ifnextchar[{\@Acrfull{hyper=false,#1}{#2}}%
3471 {\@Acrfull{hyper=false,#1}{#2}[]}%
3472 }
3473 \newcommand*\ns@Acrfull[2][]{%
3474 \new@ifnextchar[{\@Acrfull{#1}{#2}}%
3475 {\@Acrfull{#1}{#2}[]}%
3476 }

Low-level macro:
3477 \def\@Acrfull#1#2[#3]{%
3478 \acrlinkfullformat{\@Acrlong}{\@acrshort}{#1}{#2}{#3}%
3479 }
```

`\ACRfull`

```
3480 \newrobustcmd*{\ACRfull}{%
3481 \@ifstar\s@ACRfull\ns@ACRfull
3482 }

3483 \newcommand*\s@ACRfull[2][]{%
3484 \new@ifnextchar[{\@ACRfull{hyper=false,#1}{#2}}%
3485 {\@ACRfull{hyper=false,#1}{#2}[]}%
3486 }
3487 \newcommand*\ns@ACRfull[2][]{%
3488 \new@ifnextchar[{\@ACRfull{#1}{#2}}%
3489 {\@ACRfull{#1}{#2}[]}%
3490 }

Low-level macro:
3491 \def\@ACRfull#1#2[#3]{%
3492 \acrlinkfullformat{\@ACRlong}{\@ACRshort}{#1}{#2}{#3}%
3493 }
```

Plural:

`\acrfullpl`

```
3494 \newrobustcmd*{\acrfullpl}{%
3495 \@ifstar\s@acrfullpl\ns@acrfullpl
3496 }

3497 \newcommand*\s@acrfullpl[2][]{%
3498 \new@ifnextchar[{\@acrfullpl{hyper=false,#1}{#2}}%
3499 {\@acrfullpl{hyper=false,#1}{#2}[]}%
3500 }
3501 \newcommand*\ns@acrfullpl[2][]{%
3502 \new@ifnextchar[{\@acrfullpl{#1}{#2}}%

```

```

3503             {\@acrfullpl{#1}{#2} []}%
3504 }

```

Low-level macro:

```

3505 \def\@acrfullpl#1#2[#3]{%
3506 \acrlinkfullformat{\@acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%
3507 }

```

\Acrfullpl

```

3508 \newrobustcmd*\@Acrfullpl}{%
3509 \@ifstar\s@Acrfullpl\@ns@Acrfullpl
3510 }

3511 \newcommand*\s@Acrfullpl[2] []{%
3512 \new@ifnextchar[{\@Acrfullpl{hyper=false,#1}{#2}}%
3513             {\@Acrfullpl{hyper=false,#1}{#2} []}%
3514 }

3515 \newcommand*\ns@Acrfullpl[2] []{%
3516 \new@ifnextchar[{\@Acrfullpl{#1}{#2}}%
3517             {\@Acrfullpl{#1}{#2} []}%
3518 }

```

Low-level macro:

```

3519 \def\@Acrfullpl#1#2[#3]{%
3520 \acrlinkfullformat{\@Acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%
3521 }

```

\ACRfullpl

```

3522 \newrobustcmd*\@ACRfullpl}{%
3523 \@ifstar\s@ACRfullpl\@ns@ACRfullpl
3524 }

3525 \newcommand*\s@ACRfullpl[2] []{%
3526 \new@ifnextchar[{\@ACRfullpl{hyper=false,#1}{#2}}%
3527             {\@ACRfullpl{hyper=false,#1}{#2} []}%
3528 }

3529 \newcommand*\ns@ACRfullpl[2] []{%
3530 \new@ifnextchar[{\@ACRfullpl{#1}{#2}}%
3531             {\@ACRfullpl{#1}{#2} []}%
3532 }

```

Low-level macro:

```

3533 \def\@ACRfullpl#1#2[#3]{%
3534 \acrlinkfullformat{\@ACRlongpl}{\@ACRshortpl}{#1}{#2}{#3}%
3535 }

```

1.17 Predefined acronym styles

\acronymfont This is only used with the additional acronym styles:

```

3536 \newcommand{\acronymfont}[1]{#1}

```

`\firstacronymfont` This is only used with the additional acronym styles:
 3537 `\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.
 3538 `\newcommand*{\acrnameformat}[2]{\acronymfont{#1}}`

Define some tokens used by `\newacronym`:

`\glskeylisttok`
 3539 `\newtoks\glskeylisttok`

`\glslabeltok`
 3540 `\newtoks\glslabeltok`

`\glsshorttok`
 3541 `\newtoks\glsshorttok`

`\glslongtok`
 3542 `\newtoks\glslongtok`

`\newacronymhook` Provide a hook for `\newacronym`:
 3543 `\newcommand*{\newacronymhook}{}`

`\SetDefaultAcronymDisplayStyle` Sets the default acronym display style for given glossary.
 3544 `\newcommand*{\SetDefaultAcronymDisplayStyle}[1]{%`
 3545 `\defglsdisplay[#1]{##1##4}%`
 3546 `\defglsdisplayfirst[#1]{##1##4}%`
 3547 `}`

`\DefaultNewAcronymDef` Sets up the acronym definition for the default style. The information is provided
 by the tokens `\glslabeltok`, `\glsshorttok`, `\glslongtok` and `\glskeylisttok`.
 3548 `\newcommand*{\DefaultNewAcronymDef}{%`
 3549 `\edef\@do@newglossaryentry{%`
 3550 `\noexpand\newglossaryentry{\the\glslabeltok}%`
 3551 `{%`
 3552 `type=\acronymtype,%`
 3553 `name={\the\glsshorttok},%`
 3554 `sort={\the\glsshorttok},%`
 3555 `text={\the\glsshorttok},%`
 3556 `first={\acrfullformat{\the\glslongtok}{\the\glsshorttok}},%`
 3557 `plural={\the\glsshorttok\noexpand\acrpluralsuffix},%`
 3558 `firstplural={\acrfullformat{\noexpand\@glo@longpl}%`
 3559 `{\noexpand\@glo@shortpl}},%`
 3560 `short={\the\glsshorttok},%`
 3561 `shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%`
 3562 `long={\the\glslongtok},%`
 3563 `longplural={\the\glslongtok\noexpand\acrpluralsuffix},%`
 3564 `description={\the\glslongtok},%`
 3565 `descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%`

Remaining options specified by the user:

```
3566     \the\glskeylisttok
3567     }%
3568     }%
3569     \do@newglossaryentry
3570 }
```

`\SetDefaultAcronymStyle` Set up the default acronym style:

```
3571 \newcommand*\SetDefaultAcronymStyle{%
```

Set the display style:

```
3572   \@for\@gls@type:=\@glsacronymlists\do{%
3573     \SetDefaultAcronymDisplayStyle{\@gls@type}%
3574   }%
```

Set up the definition of `\newacronym`:

```
3575 \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).

```
3576   \ifx\@glsacronymlists\@empty
3577     \def\@glo@type{\acronymtype}%
3578     \setkeys{glossentry}{##1}%
3579     \DeclareAcronymList{\@glo@type}%
3580     \SetDefaultAcronymDisplayStyle{\@glo@type}%
3581     \fi
3582     \glskeylisttok{##1}%
3583     \glslabeltok{##2}%
3584     \glsshorttok{##3}%
3585     \glslongtok{##4}%
3586     \newacronymhook
3587     \DefaultNewAcronymDef
3588   }%
3589 \renewcommand*\acrpluralsuffix{\glspluralsuffix}%
3590 }
```

`\acrfootnote` Used by the footnote acronym styles.

```
3591 \newcommand*\acrfootnote}[3]{\acrlinkfootnote{#1}{#2}{#3}}
```

`\acrlinkfootnote`

```
3592 \newcommand*\acrlinkfootnote}[3]{%
3593   \footnote{\glslink[#1]{#2}{#3}}%
3594 }
```

`\acrnofootnote`

```
3595 \newcommand*\acrnofootnote}[3]{%
3596   \footnote{#3}%
3597 }
```

`tionFootnoteAcronymDisplayStyle` Sets the acronym display style for given glossary for the description and footnote combination.

```

3598 \newcommand*{\SetDescriptionFootnoteAcronymDisplayStyle}[1]{%
3599   \defglsdisplayfirst[#1]{%
3600     \firstacronymfont{##1}##4%
3601     \expandafter\protect\expandafter\acrfootnote\expandafter
3602       {\@gls@link@opts}{\@gls@link@label}{##3}
3603   }%
3604   \defglsdisplay[#1]{\acronymfont{##1}##4}%
3605 }

```

`escriptionFootnoteNewAcronymDef`

```

3606 \newcommand*{\DescriptionFootnoteNewAcronymDef}{%
3607   \edef\@do@newglossaryentry{%
3608     \noexpand\newglossaryentry{\the\glslabeltok}%
3609     {%
3610       type=\acronymtype,%
3611       name={\noexpand\acronymfont{\the\glsshorttok}},%
3612       sort={\the\glsshorttok},%
3613       text={\the\glsshorttok},%
3614       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3615       short={\the\glsshorttok},%
3616       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3617       long={\the\glslongtok},%
3618       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3619       symbol={\the\glslongtok},%
3620       symbolplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3621       \the\glskeylisttok
3622     }%
3623   }%
3624   \@do@newglossaryentry
3625 }

```

`DescriptionFootnoteAcronymStyle` 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.

```

3626 \newcommand*{\SetDescriptionFootnoteAcronymStyle}{%
3627   \renewcommand{\newacronym}[4][ ]{%
3628     \ifx\@glsacronymlists\@empty
3629       \def\@glo@type{\acronymtype}%
3630       \setkeys{glossentry}{##1}%
3631       \DeclareAcronymList{\@glo@type}%
3632       \SetDescriptionFootnoteAcronymDisplayStyle{\@glo@type}%
3633     \fi
3634     \glskeylisttok{##1}%
3635     \glslabeltok{##2}%
3636     \glsshorttok{##3}%
3637     \glslongtok{##4}%

```

```

3638 \newacronymhook
3639 \DescriptionFootnoteNewAcronymDef
3640 }%

```

If footnote package option is specified, set the first use to append the long form (stored in symbol) as a footnote.

```

3641 \@for\@gls@type:=\@glsacronymlists\do{%
3642   \SetDescriptionFootnoteAcronymDisplayStyle{\@gls@type}%
3643 }%

```

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.

```

3644 \ifglsacrsmallcaps
3645   \renewcommand*\acronymfont}[1]{\textsc{##1}}%
3646   \renewcommand*\acrpluralsuffix{%
3647     \textup{glspluralsuffix}}%
3648 \else
3649   \ifglsacrsmaller
3650     \renewcommand*\acronymfont}[1]{\textsmaller{##1}}%
3651   \fi
3652 \fi

```

Check for package option clash

```

3653 \ifglsacrdua
3654   \PackageError{glossaries}{Option clash: 'footnote' and 'dua'
3655     can't both be set}{}%
3656 \fi
3657 }%

```

`\SetDescriptionDUAAcronymDisplayStyle` Sets the acronym display style for given glossary with description and dua combination.

```

3658 \newcommand*\SetDescriptionDUAAcronymDisplayStyle}[1]{%
3659   \defglsdisplay[#1]{##1##4}%
3660   \defglsdisplayfirst[#1]{##1##4}%
3661 }

```

`\DescriptionDUANewAcronymDef`

```

3662 \newcommand*\DescriptionDUANewAcronymDef}{%
3663   \edef\@do@newglossaryentry{%
3664     \noexpand\newglossaryentry{\the\glslabeltok}%
3665     {%
3666       type=\acronymtype,%
3667       name={\the\glslongtok},%
3668       sort={\the\glslongtok},%
3669       text={\the\glslongtok},%
3670       plural={\the\glslongtok\noexpand\acrpluralsuffix},%
3671       short={\the\glsshorttok},%
3672       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3673       long={\the\glslongtok},%

```

```

3674     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3675     symbol={\the\glsshorttok},%
3676     symbolplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3677     \the\glskeylisttok
3678   }%
3679 }%
3680 \do@newglossaryentry
3681 }

```

`\SetDescriptionDUAAcronymStyle` 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.

```

3682 \newcommand*\SetDescriptionDUAAcronymStyle{%
3683   \ifglsacrsmallcaps
3684     \PackageError{glossaries}{Option clash: 'smallcaps' and 'dua'
3685       can't both be set}{}%
3686   \else
3687     \ifglsacrsmaller
3688       \PackageError{glossaries}{Option clash: 'smaller' and 'dua'
3689         can't both be set}{}%
3690     \fi
3691   \fi
3692   \renewcommand{\newacronym}[4][]{%
3693     \ifx\@glsacronymlists\@empty
3694       \def\@glo@type{\acronymtype}%
3695       \setkeys{glossentry}{##1}%
3696       \DeclareAcronymList{\@glo@type}%
3697       \SetDescriptionDUAAcronymDisplayStyle{\@glo@type}%
3698     \fi
3699     \glskeylisttok{##1}%
3700     \glslabeltok{##2}%
3701     \glsshorttok{##3}%
3702     \glslongtok{##4}%
3703     \newacronymhook
3704     \DescriptionDUANewAcronymDef
3705   }%

```

Set display.

```

3706   \@for\@gls@type:=\@glsacronymlists\do{%
3707     \SetDescriptionDUAAcronymDisplayStyle{\@gls@type}%
3708   }%
3709 }%

```

`\SetDescriptionAcronymDisplayStyle` Sets the acronym display style for given glossary using the description setting (but not footnote or dua).

```

3710 \newcommand*\SetDescriptionAcronymDisplayStyle}[1]{%
3711   \defglsdisplayfirst[#1]{%
3712     ##1##4 (\firstacronymfont{##3})}%
3713   \defglsdisplay[#1]{\acronymfont{##1}##4}%
3714 }

```

`\DescriptionNewAcronymDef`

```
3715 \newcommand*{\DescriptionNewAcronymDef}{%
3716   \edef\@do@newglossaryentry{%
3717     \noexpand\newglossaryentry{\the\glslabeltok}%
3718     {%
3719       type=\acronymtype,%
3720       name={\noexpand
3721         \acronymformat{\the\glsshorttok}{\the\glslongtok}},%
3722       sort={\the\glsshorttok},%
3723       first={\the\glslongtok},%
3724       firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3725       text={\the\glsshorttok},%
3726       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3727       short={\the\glsshorttok},%
3728       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3729       long={\the\glslongtok},%
3730       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3731       symbol={\noexpand\@glo@text},%
3732       symbolplural={\noexpand\@glo@plural},%
3733       \the\glskeylisttok}%
3734   }%
3735   \@do@newglossaryentry
3736 }
```

`\SetDescriptionAcronymStyle` 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 `\acronymformat` to allow the user to override the way the name is displayed in the list of acronyms.

```
3737 \newcommand*{\SetDescriptionAcronymStyle}{%
3738   \renewcommand{\newacronym}[4][]{%
3739     \ifx\@glsacronymlists\@empty
3740       \def\@glo@type{\acronymtype}%
3741       \setkeys{glossentry}{##1}%
3742       \DeclareAcronymList{\@glo@type}%
3743       \SetDescriptionAcronymDisplayStyle{\@glo@type}%
3744       \fi
3745       \glskeylisttok{##1}%
3746       \glslabeltok{##2}%
3747       \glsshorttok{##3}%
3748       \glslongtok{##4}%
3749       \newacronymhook
3750       \DescriptionNewAcronymDef
3751   }%
```

Set display.

```
3752   \@for\@gls@type:=\@glsacronymlists\do{%
3753     \SetDescriptionAcronymDisplayStyle{\@gls@type}%
3754   }%
```

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.

```
3755 \ifglsacrsmallcaps
3756   \renewcommand{\acronymfont}[1]{\textsc{##1}}
3757   \renewcommand*{\acrpluralsuffix}{%
3758     \textup{\glspluralsuffix}}%
3759 \else
3760   \ifglsacrsmaller
3761     \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%
3762   \fi
3763 \fi
3764 }%
```

\SetFootnoteAcronymDisplayStyle Sets the acronym display style for given glossary with footnote setting (but not description or dua).

```
3765 \newcommand*{\SetFootnoteAcronymDisplayStyle}[1]{%
3766   \defglsdisplayfirst[#1]{%
3767     \firstacronymfont{##1}##4%
3768     \expandafter\protect\expandafter\acrfootnote\expandafter
3769       {\@gls@link@opts}{\@gls@link@label}{##2}%
3770   }%
3771   \defglsdisplay[#1]{\acronymfont{##1}##4}%
3772 }
```

\FootnoteNewAcronymDef

```
3773 \newcommand*{\FootnoteNewAcronymDef}{%
3774   \edef\@do@newglossaryentry{%
3775     \noexpand\newglossaryentry{\the\glslabeltok}%
3776     {%
3777       type=\acronymtype,%
3778       name={\noexpand\acronymfont{\the\glsshorttok}},%
3779       sort={\the\glsshorttok},%
3780       text={\the\glsshorttok},%
3781       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3782       short={\the\glsshorttok},%
3783       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3784       long={\the\glslongtok},%
3785       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3786       description={\the\glslongtok},%
3787       descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3788       \the\glskeylisttok
3789     }%
3790   }%
3791   \@do@newglossaryentry
3792 }
```

\SetFootnoteAcronymStyle 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.

```
3793 \newcommand*{\SetFootnoteAcronymStyle}{%
3794   \renewcommand{\newacronym}[4] []{%
```

```

3795 \ifx\@glsacronymlists\@empty
3796 \def\@glo@type{\acronymtype}%
3797 \setkeys{glossentry}{##1}%
3798 \DeclareAcronymList{\@glo@type}%
3799 \SetFootnoteAcronymDisplayStyle{\@glo@type}%
3800 \fi
3801 \glskeylisttok{##1}%
3802 \glslabeltok{##2}%
3803 \glsshorttok{##3}%
3804 \glslongtok{##4}%
3805 \newacronymhook
3806 \FootnoteNewAcronymDef
3807 }%

```

Set display

```

3808 \@for\@gls@type:=\@glsacronymlists\do{%
3809 \SetFootnoteAcronymDisplayStyle{\@gls@type}%
3810 }%

```

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.

```

3811 \ifglsacrsmallcaps
3812 \renewcommand*\acronymfont[1]{\textsc{##1}}%
3813 \renewcommand*\acrpluralsuffix{%
3814 \textup{\glspluralsuffix}}%
3815 \else
3816 \ifglsacrsmaller
3817 \renewcommand*\acronymfont[1]{\textsmaller{##1}}%
3818 \fi
3819 \fi

```

Check for option clash

```

3820 \ifglsacrdua
3821 \PackageError{glossaries}{Option clash: 'footnote' and 'dua'
3822 can't both be set}{}%
3823 \fi
3824 }%

```

`\SetSmallAcronymDisplayStyle` Sets the acronym display style for given glossary where neither footnote nor description is required, but smallcaps or smaller specified.

```

3825 \newcommand*\SetSmallAcronymDisplayStyle[1]{%
3826 \defglsdisplayfirst[#1]{##1##4 (\firstacronymfont{##3})}
3827 \defglsdisplay[#1]{\acronymfont{##1}##4}%
3828 }

```

`\SmallNewAcronymDef`

```

3829 \newcommand*\SmallNewAcronymDef{%
3830 \edef\@do@newglossaryentry{%
3831 \noexpand\newglossaryentry{\the\glslabeltok}%

```

```

3832   {%
3833     type=\acronymtype,%
3834     name={\noexpand\acronymfont{\the\glsshorttok}},%
3835     sort={\the\glsshorttok},%
3836     text={\noexpand\@glo@symbol},%
3837     plural={\noexpand\@glo@symbolplural},%
3838     first={\the\glslongtok},%
3839     firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3840     short={\the\glsshorttok},%
3841     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3842     long={\the\glslongtok},%
3843     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3844     description={\noexpand\@glo@first},%
3845     descriptionplural={\noexpand\@glo@firstplural},%
3846     symbol={\the\glsshorttok},%
3847     symbolplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3848     \the\glskeylisttok
3849   }%
3850 }%
3851 \do@newglossaryentry
3852 }

```

`\SetSmallAcronymStyle` Neither footnote nor description required, but smallcaps or smaller specified. Use the symbol key to store the short form and first to store the long form.

```

3853 \newcommand*{\SetSmallAcronymStyle}{%
3854   \renewcommand{\newacronym}[4][]{%
3855     \ifx\@glsacronymlists\@empty
3856       \def\@glo@type{\acronymtype}%
3857       \setkeys{glossentry}{##1}%
3858       \DeclareAcronymList{\@glo@type}%
3859       \SetSmallAcronymDisplayStyle{\@glo@type}%
3860       \fi
3861       \glskeylisttok{##1}%
3862       \glslabeltok{##2}%
3863       \glsshorttok{##3}%
3864       \glslongtok{##4}%
3865       \newacronymhook
3866       \SmallNewAcronymDef
3867   }%

```

Change the display since first only contains long form.

```

3868   \@for\@gls@type:=\@glsacronymlists\do{%
3869     \SetSmallAcronymDisplayStyle{\@gls@type}%
3870   }%

```

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.

```

3871   \ifglsacrsmallcaps
3872     \renewcommand*{\acronymfont}[1]{\textsc{##1}}

```

```

3873   \renewcommand*\acrpluralsuffix}{%
3874       \textup{\glspluralsuffix}}%
3875   \else
3876       \renewcommand*\acronymfont}[1]{\textsmaller{##1}}
3877   \fi

    check for option clash
3878   \ifglsacrdua
3879       \ifglsacrsmallcaps
3880           \PackageError{glossaries}{Option clash: ‘smallcaps’ and ‘dua’
3881               can’t both be set}{}%
3882       \else
3883           \PackageError{glossaries}{Option clash: ‘smaller’ and ‘dua’
3884               can’t both be set}{}%
3885       \fi
3886   \fi
3887 }%

```

`\SetDUADisplayStyle` Sets the acronym display style for given glossary with dua setting.

```

3888 \newcommand*\SetDUADisplayStyle}[1]{%
3889     \defglsdisplay[#1]{##1##4}%
3890     \defglsdisplayfirst[#1]{##1##4}%
3891 }

```

`\DUANewAcronymDef`

```

3892 \newcommand*\DUANewAcronymDef}{%
3893     \edef\@do@newglossaryentry{%
3894         \noexpand\newglossaryentry{\the\glslabeltok}%
3895         {%
3896             type=\acronymtype,%
3897             name={\the\glsshorttok},%
3898             text={\the\glslongtok},%
3899             plural={\the\glslongtok\noexpand\acrpluralsuffix},%
3900             short={\the\glsshorttok},%
3901             shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3902             long={\the\glslongtok},%
3903             longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3904             description={\the\glslongtok},%
3905             symbol={\the\glsshorttok},%
3906             symbolplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3907             \the\glskeylisttok
3908         }%
3909     }%
3910     \@do@newglossaryentry
3911 }

```

`\SetDUASStyle` Always expand acronyms.

```

3912 \newcommand*\SetDUASStyle}{%
3913     \renewcommand{\newacronym}[4][]{%
3914         \ifx\@glsacronymlists\@empty

```

```

3915     \def\@glo@type{\acronymtype}%
3916     \setkeys{glossentry}{##1}%
3917     \DeclareAcronymList{\@glo@type}%
3918     \SetDUADisplayStyle{\@glo@type}%
3919     \fi
3920     \glskeylisttok{##1}%
3921     \glslabeltok{##2}%
3922     \glsshorttok{##3}%
3923     \glslongtok{##4}%
3924     \newacronymhook
3925     \DUANewAcronymDef
3926 }%
    Set the display
3927 \@for\@gls@type:=\@glsacronymlists\do{%
3928     \SetDUADisplayStyle{\@gls@type}%
3929 }%
3930 }

```

\SetAcronymStyle

```

3931 \newcommand*{\SetAcronymStyle}{%
3932     \SetDefaultAcronymStyle
3933     \ifglsacrdescription
3934         \ifglsacrfootnote
3935             \SetDescriptionFootnoteAcronymStyle
3936         \else
3937             \ifglsacrdua
3938                 \SetDescriptionDUAAcronymStyle
3939             \else
3940                 \SetDescriptionAcronymStyle
3941             \fi
3942         \fi
3943     \else
3944         \ifglsacrfootnote
3945             \SetFootnoteAcronymStyle
3946         \else
3947             \ifthenelse{\boolean{glsacrsmallcaps}\OR
3948                 \boolean{glsacrsmaller}}{%
3949                 {%
3950                     \SetSmallAcronymStyle
3951                 }%
3952             }%
3953         \ifglsacrdua
3954             \SetDUASStyle
3955         \fi
3956     }%
3957     \fi
3958     \fi
3959 }

```

Set the acronym style according to the package options

3960 `\SetAcronymStyle`

Allow user to define their own custom acronyms. (For compatibility with versions before v3.0, the short form is stored in the user1 key, the plural short form is stored in the user2 key, the long form is stored in the user3 key and the plural long form is stored in the user4 key.) Defaults to displaying only the acronym with the long form as the description.

`\SetCustomDisplayStyle` Sets the acronym display style.

```
3961 \newcommand*\SetCustomDisplayStyle[1]{%
3962   \defglsdisplay[#1]{##1##4}%
3963   \defglsdisplayfirst[#1]{##1##4}%
3964 }
```

`\CustomAcronymFields`

```
3965 \newcommand*\CustomAcronymFields{%
3966   name={\the\glsshorttok},%
3967   description={\the\glslongtok},%
3968   first={\noexpand\acrfullformat{\the\glslongtok}{\the\glsshorttok}},%
3969   firstplural={\noexpand\acrfullformat
3970     {\the\glslongtok\noexpand\acrpluralsuffix}{\the\glsshorttok}},%
3971   text={\the\glsshorttok},%
3972   plural={\the\glsshorttok\noexpand\acrpluralsuffix}%
3973 }
```

`\CustomNewAcronymDef`

```
3974 \newcommand*\CustomNewAcronymDef{%
3975   \protected@edef\@do@newglossaryentry{%
3976     \noexpand\newglossaryentry{\the\glslabeltok}%
3977     {%
3978       type=\acronymtype,%
3979       short={\the\glsshorttok},%
3980       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3981       long={\the\glslongtok},%
3982       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3983       user1={\the\glsshorttok},%
3984       user2={\the\glsshorttok\noexpand\acrpluralsuffix},%
3985       user3={\the\glslongtok},%
3986       user4={\the\glslongtok\noexpand\acrpluralsuffix},%
3987       \CustomAcronymFields,%
3988       \the\glskeylisttok
3989     }%
3990   }%
3991   \@do@newglossaryentry
3992 }
```

`\SetCustomStyle`

```
3993 \newcommand*\SetCustomStyle{%
3994   \renewcommand{\newacronym}[4] []{%
```

```

3995 \ifx\@glsacronymlists\@empty
3996 \def\@glo@type{\acronymtype}%
3997 \setkeys{glossentry}{##1}%
3998 \DeclareAcronymList{\@glo@type}%
3999 \SetCustomDisplayStyle{\@glo@type}%
4000 \fi
4001 \glskeylisttok{##1}%
4002 \glslabeltok{##2}%
4003 \glsshorttok{##3}%
4004 \gslongtok{##4}%
4005 \newacronymhook
4006 \CustomNewAcronymDef
4007 }%

Set the display
4008 \@for\@gls@type:=\@glsacronymlists\do{%
4009 \SetCustomDisplayStyle{\@gls@type}%
4010 }%
4011 }

```

\DefineAcronymSynonyms

```
4012 \newcommand*{\DefineAcronymSynonyms}{%
```

Short form

\acs

```
4013 \let\acs\acrshort
```

First letter uppercase short form

\Acs

```
4014 \let\Acs\Acrshort
```

Plural short form

\acsp

```
4015 \let\acsp\acrshortpl
```

First letter uppercase plural short form

\Acsp

```
4016 \let\Acsp\Acrshortpl
```

Long form

\acl

```
4017 \let\acl\aclong
```

Plural long form

\aclp

```
4018 \let\aclp\aclongpl
```

First letter upper case long form

```
\Ac1
4019 \let\Ac1\Acrlong
```

First letter upper case plural long form

```
\Ac1p
4020 \let\Ac1p\Acrlongpl
```

Full form

```
\acf
4021 \let\acf\acrfull
```

Plural full form

```
\acfp
4022 \let\acfp\acrfullpl
```

First letter upper case full form

```
\Acf
4023 \let\Acf\Acrfull
```

First letter upper case plural full form

```
\Acfp
4024 \let\Acfp\Acrfullpl
```

Standard form

```
\ac
4025 \let\ac\gls
```

First upper case standard form

```
\Ac
4026 \let\Ac\Gls
```

Standard plural form

```
\acp
4027 \let\acp\glspl
```

Standard first letter upper case plural form

```
\Acp
4028 \let\Acp\Glspl
4029 }
```

Define synonyms if required

```
4030 \ifglsacrshortcuts
4031 \DefineAcronymSynonyms
4032 \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.

```
4033 \RequirePackage{glossary-hypernav}
```

The styles that use list-like environments. These are not loaded if the `nolist` option is used:

```
4034 \@gls@loadlist
```

The styles that use the `longtable` environment. These are not loaded if the `nolong` package option is used.

```
4035 \@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.

```
4036 \@gls@loadsuper
```

The tree-like styles. These are not loaded if the `notree` package option is used.

```
4037 \@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.

```
4038 \ifx\@glossary@default@style\relax
```

```
4039 \else
```

```
4040 \glossarystyle{\@glossary@default@style}
```

```
4041 \fi
```

1.19 Debugging Commands

```
\showgloparent \showgloparent{\label}
```

```
4042 \newcommand*\showgloparent}[1]{%
```

```
4043 \expandafter\show\csname glo@#1@parent\endcsname
```

```
4044 }
```

```
\showglolevel \showglolevel{\label}
```

```
4045 \newcommand*\showglolevel}[1]{%
```

```
4046 \expandafter\show\csname glo@#1@level\endcsname
```

```
4047 }
```

```
\showglotext \showglotext{\label}
```

```
4048 \newcommand*{\showglotext}[1]{%
4049 \expandafter\show\csname glo@#1@text\endcsname
4050 }
```

`\showgloplural` `\showgloplural{<label>}`

```
4051 \newcommand*{\showgloplural}[1]{%
4052 \expandafter\show\csname glo@#1@plural\endcsname
4053 }
```

`\showglofirst` `\showglofirst{<label>}`

```
4054 \newcommand*{\showglofirst}[1]{%
4055 \expandafter\show\csname glo@#1@first\endcsname
4056 }
```

`\showglofirstpl` `\showglofirstpl{<label>}`

```
4057 \newcommand*{\showglofirstpl}[1]{%
4058 \expandafter\show\csname glo@#1@firstpl\endcsname
4059 }
```

`\showglotype` `\showglotype{<label>}`

```
4060 \newcommand*{\showglotype}[1]{%
4061 \expandafter\show\csname glo@#1@type\endcsname
4062 }
```

`\showglocounter` `\showglocounter{<label>}`

```
4063 \newcommand*{\showglocounter}[1]{%
4064 \expandafter\show\csname glo@#1@counter\endcsname
4065 }
```

`\showglouser` `\showglouser{<label>}`

```
4066 \newcommand*{\showglouser}[1]{%
4067 \expandafter\show\csname glo@#1@user\endcsname
4068 }
```

```
\showglouserii \showglouserii{<label>}
```

```
4069 \newcommand*{\showglouserii}[1]{%  
4070 \expandafter\show\csname glo@#1@userii\endcsname  
4071 }
```

```
\showglouseriii \showglouseriii{<label>}
```

```
4072 \newcommand*{\showglouseriii}[1]{%  
4073 \expandafter\show\csname glo@#1@useriii\endcsname  
4074 }
```

```
\showglouseriv \showglouseriv{<label>}
```

```
4075 \newcommand*{\showglouseriv}[1]{%  
4076 \expandafter\show\csname glo@#1@useriv\endcsname  
4077 }
```

```
\showglouserv \showglouserv{<label>}
```

```
4078 \newcommand*{\showglouserv}[1]{%  
4079 \expandafter\show\csname glo@#1@userv\endcsname  
4080 }
```

```
\showglouservi \showglouservi{<label>}
```

```
4081 \newcommand*{\showglouservi}[1]{%  
4082 \expandafter\show\csname glo@#1@uservi\endcsname  
4083 }
```

```
\showgloname \showgloname{<label>}
```

```
4084 \newcommand*{\showgloname}[1]{%  
4085 \expandafter\show\csname glo@#1@name\endcsname  
4086 }
```

```
\showglodesc \showglodesc{<label>}
```

```
4087 \newcommand*{\showlodesc}[1]{%
4088   \expandafter\show\csname glo@#1@desc\endcsname
4089 }
```

`\showlodescplural` `\showlodescplural{<label>}`

```
4093 \newcommand*{\showlodescplural}[1]{%
4094   \expandafter\show\csname glo@#1@descplural\endcsname
4095 }
```

`\showglosort` `\showglosort{<label>}`

```
4093 \newcommand*{\showglosort}[1]{%
4094   \expandafter\show\csname glo@#1@sort\endcsname
4095 }
```

`\showglosymbol` `\showglosymbol{<label>}`

```
4096 \newcommand*{\showglosymbol}[1]{%
4097   \expandafter\show\csname glo@#1@symbol\endcsname
4098 }
```

`\showglosymbolplural` `\showglosymbolplural{<label>}`

```
4099 \newcommand*{\showglosymbolplural}[1]{%
4100   \expandafter\show\csname glo@#1@symbolplural\endcsname
4101 }
```

`\showgloindex` `\showgloindex{<label>}`

```
4102 \newcommand*{\showgloindex}[1]{%
4103   \expandafter\show\csname glo@#1@index\endcsname
4104 }
```

`\showgloflag` `\showgloflag{<label>}`

```
4105 \newcommand*{\showgloflag}[1]{%
4106   \expandafter\show\csname ifglo@#1@flag\endcsname
4107 }
```

`\showacronymlists` `\showacronymlists`

Show list of glossaries that have been flagged as a list of acronyms.

```
4108 \newcommand*{\showacronymlists}{%
4109   \show\@glsacronymlists
4110 }
```

`\showglossaries` `\showglossaries`

Show list of defined glossaries.

```
4111 \newcommand*{\showglossaries}{%
4112   \show\@glo@types
4113 }
```

`\showglossaryin` `\showglossaryin{<glossary-label>}`

Show the ‘in’ extension for the given glossary.

```
4114 \newcommand*{\showglossaryin}[1]{%
4115   \expandafter\show\csname @glo@type@#1@in\endcsname
4116 }
```

`\showglossaryout` `\showglossaryout{<glossary-label>}`

Show the ‘out’ extension for the given glossary.

```
4117 \newcommand*{\showglossaryout}[1]{%
4118   \expandafter\show\csname @glo@type@#1@out\endcsname
4119 }
```

`\showglossarytitle` `\showglossarytitle{<glossary-label>}`

Show the title for the given glossary.

```
4120 \newcommand*{\showglossarytitle}[1]{%
4121   \expandafter\show\csname @glo@type@#1@title\endcsname
4122 }
```

`\showglossarycounter` `\showglossarycounter{<glossary-label>}`

Show the counter for the given glossary.

```
4123 \newcommand*{\showglossarycounter}[1]{%
4124   \expandafter\show\csname @glo@type@#1@counter\endcsname
4125 }
```

```
\showglossaryentries \showglossaryentries{<glossary-label>}
```

Show the list of entry labels for the given glossary.

```
4126 \newcommand*{\showglossaryentries}[1]{%
4127 \expandafter\show\csname @glolist@#1\endcsname
4128 }
```

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.

```
4129 \csname ifglsc-compatible-2.07\endcsname
4130 \RequirePackage{glossaries-compatible-207}
4131 \fi
```

2 Mfirstuc Documented Code

```
4132 \NeedsTeXFormat{LaTeX2e}
4133 \ProvidesPackage{mfirstuc}[2011/04/02 v1.05 (NLCT)]
```

`\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`.

```
4134 \newif\if@glscs
4135 \newtoks\@glsmfirst
4136 \newtoks\@glsmrest
4137 \def\makefirstuc#1{%
4138 \def\gls@argi{#1}%
4139 \ifx\gls@argi\@empty
```

If the argument is empty, do nothing.

```
4140 \else
```

```

4141 \def\@gls@tmp{\ #1}%
4142 \@onelevel@sanitize\@gls@tmp
4143 \expandafter\@gls@checkcs\@gls@tmp\relax\relax
4144 \if@glscs
4145 \@gls@getbody #1{\@nil
4146 \ifx\@gls@rest\@empty
4147 \glsmakefirstuc{#1}%
4148 \else
4149 \expandafter\@gls@split\@gls@rest\@nil
4150 \ifx\@gls@first\@empty
4151 \glsmakefirstuc{#1}%
4152 \else
4153 \expandafter\@glsmfirst\expandafter{\@gls@first}%
4154 \expandafter\@glsmrest\expandafter{\@gls@rest}%
4155 \edef\@gls@domfirstuc{\noexpand\@gls@body
4156 \noexpand\glsmakefirstuc\the\@glsmfirst}%
4157 \the\@glsmrest}%
4158 \@gls@domfirstuc
4159 \fi
4160 \fi
4161 \else
4162 \glsmakefirstuc{#1}%
4163 \fi
4164 \fi
4165 }

```

Put first argument in \@gls@first and second argument in \@gls@rest:

```

4166 \def\@gls@split#1#2\@nil{%
4167 \def\@gls@first{#1}\def\@gls@rest{#2}%
4168 }

4169 \def\@gls@checkcs#1 #2#3\relax{%
4170 \def\@gls@argi{#1}\def\@gls@argii{#2}%
4171 \ifx\@gls@argi\@gls@argii
4172 \@glscstrue
4173 \else
4174 \@glscsfalse
4175 \fi
4176 }

```

Make first thing upper case:

```

4177 \def\@gls@makefirstuc#1{\MakeUppercase #1}

```

\glsmakefirstuc Provide a user command to make it easier to customise.

```

4178 \newcommand*{\glsmakefirstuc}[1]{\@gls@makefirstuc{#1}}

```

Get the first grouped argument and stores in \@gls@body.

```

4179 \def\@gls@getbody#1#{\def\@gls@body{#1}\@gls@gobbletonil}

```

Scoup up everything to \@nil and store in \@gls@rest:

```

4180 \def\@gls@gobbletonil#1\@nil{\def\@gls@rest{#1}}

```

`\xmakefirstuc` Expand argument once before applying `\makefirstuc` (added v1.01).

```
4181 \newcommand*\xmakefirstuc}[1]{%
4182 \expandafter\makefirstuc\expandafter{#1}}
```

3 Glossary Styles

3.1 Glossary hyper-navigation definitions (glossary-hypernav package)

Package Definition:

```
4183 \ProvidesPackage{glossary-hypernav}[2007/07/04 v1.01 (NLCT)]
```

The commands defined in this package are provided to help navigate around the groups within a glossary (see [subsection 1.15.](#)) `\printglossary` (and `\printglossaries`) set `\glo@type` to the label of the current glossary. This is used to create a unique hypertarget in the event of multiple glossaries.

```
\glsnavhyperlink[<type>]{<label>}{<text>}
```

This command makes `<text>` a hyperlink to the glossary group whose label is given by `<label>` for the glossary given by `<type>`.

`\glsnavhyperlink`

```
4184 \newcommand*\glsnavhyperlink}[3][\glo@type]{%
4185 \edef\gls@grplabel{#2}\protected@edef\gls@grptitle{#3}%
4186 \@glslink{glsn:#1@#2}{#3}}
```

```
\glsnavhypertarget[<type>]{<label>}{<text>}
```

This command makes `<text>` a hypertarget for the glossary group whose label is given by `<label>` in the glossary given by `<type>`. If `<type>` is omitted, `\glo@type` is used which is set by `\printglossary` to the current glossary label.

`\glsnavhypertarget`

```
4187 \newcommand*\glsnavhypertarget}[3][\glo@type]{%
  Add this group to the aux file for re-run check.
4188 \protected@write\auxout{\string\gls@hypergroup{#1}{#2}}%
  Add the target.
4189 \@glstarget{glsn:#1@#2}{#3}%
  Check list of know groups to determine if a re-run is required.
4190 \expandafter\let
4191 \expandafter\gls@list\csname @gls@hypergroup@#1\endcsname
  Iterate through list and terminate loop if this group is found.
4192 \@for\gls@elem:=\gls@list\do{%
4193 \ifthenelse{\equal{\gls@elem}{#2}}{\endfortrue}{}}%
```

Check if list terminated prematurely.

```
4194 \if@endfor
4195 \else
    This group was not included in the list, so issue a warning.
4196 \GlossariesWarningNoLine{Navigation panel
4197     for glossary type ‘#1’^^Jmissing group ‘#2’}%
4198 \gdef\gls@hypergrouprerun{%
4199     \GlossariesWarningNoLine{Navigation panel
4200     has changed. Rerun LaTeX}}%
4201 \fi
4202 }
```

`\gls@hypergrouprerun` Give a warning at the end if re-run required

```
4203 \let\gls@hypergrouprerun\relax
4204 \AtEndDocument{\gls@hypergrouprerun}
```

`\@gls@hypergroup` This adds to (or creates) the command `\@gls@hypergrouplist@<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.

```
4205 \newcommand*{\@gls@hypergroup}[2]{%
4206 \ifundefined{@gls@hypergrouplist@#1}{%
4207     \expandafter\xdef\csname @gls@hypergrouplist@#1\endcsname{#2}%
4208 }{%
4209     \expandafter\let\expandafter\@gls@tmp
4210     \csname @gls@hypergrouplist@#1\endcsname
4211     \expandafter\xdef\csname @gls@hypergrouplist@#1\endcsname{%
4212     \@gls@tmp,#2}%
4213 }%
4214 }
```

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`

```
4215 \newcommand*{\glsnavigation}{%
4216 \def\@gls@between{}%
4217 \ifundefined{@gls@hypergrouplist@\@gls@type}{%
4218     \def\@gls@list{}%
4219 }{%
4220     \expandafter\let\expandafter\@gls@list
4221     \csname @gls@hypergrouplist@\@gls@type\endcsname
4222 }%
4223 \@for\@gls@tmp:=\@gls@list\do{%
```

```

4224 \@gls@between
4225 \glsnavhyperlink{\@gls@tmp}{\glsgetgrouptitle{\@gls@tmp}}%
4226 \let\@gls@between\glshypernavsep%
4227 }%
4228 }

```

`\glshypernavsep` Separator for the hyper navigation bar.

```
4229 \newcommand*\glshypernavsep{\space\textbar\space}
```

The `\glsymbolnav` 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.

`\glsymbolnav`

```

4230 \newcommand*\glsymbolnav{%
4231 \glsnavhyperlink{glsymbols}{\glsgetgrouptitle{glsymbols}}%
4232 \glshypernavsep
4233 \glsnavhyperlink{glsnumbers}{\glsgetgrouptitle{glsnumbers}}%
4234 \glshypernavsep
4235 }

```

3.2 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.

```
4236 \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.

```
4237 \newglossarystyle{list}{%
```

Use `description` environment:

```

4238 \renewenvironment{theglossary}%
4239 {\begin{description}}{\end{description}}%

```

No header at the start of the environment:

```
4240 \renewcommand*\glossaryheader{}%
```

No group headings:

```
4241 \renewcommand*\glsgruppeheading}[1]{}%
```

Main (level 0) entries start a new item in the list:

```

4242 \renewcommand*\glossaryentryfield}[5]{%
4243 \item[\glsentryitem{##1}\glstarget{##1}{##2}]
4244 ##3\glspostdescription\space ##5}%

```

Sub-entries continue on the same line:

```
4245 \renewcommand*\glossarysubentryfield}[6]{%
4246   \glssubentryitem{##2}%
4247   \glstarget{##2}{\strut}##4\glspostdescription\space ##6.}%
4248 % \end{macrocode}
4249 % Add vertical space between groups:
4250 % \begin{macrocode}
4251 \renewcommand*\glsgroupskip{\indexspace}%
4252 }
```

listgroup The listgroup style is like the list style, but the glossary groups have headings.

```
4253 \newglossarystyle{listgroup}{%
  Base it on the list style:
4254 \glossarystyle{list}%
  Each group has a heading:
4255 \renewcommand*\glsgroupheading}[1]{\item[\glsgrouptitle{##1}]}
```

listhypergroup The listhypergroup style is like the listgroup style, but has a set of links to the groups at the start of the glossary.

```
4256 \newglossarystyle{listhypergroup}{%
  Base it on the list style:
4257 \glossarystyle{list}%
  Add navigation links at the start of the environment:
4258 \renewcommand*\glossaryheader}{%
4259   \item[\glsnavigation]}%
  Each group has a heading with a hypertarget:
4260 \renewcommand*\glsgroupheading}[1]{%
4261   \item[\glshnavhypertarget{##1}{\glsgrouptitle{##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.

```
4262 \newglossarystyle{altlist}{%
  Base it on the list style:
4263 \glossarystyle{list}%
  Main (level 0) entries start a new item in the list with a line break after the entry name:
4264 \renewcommand*\glossaryentryfield}[5]{%
4265   \item[\glsentryitem{##1}\glstarget{##1}{##2}]\mbox{}\newline
4266   ##3\glspostdescription\space ##5}%
  Sub-entries start a new paragraph:
4267 \renewcommand*\glossarysubentryfield}[6]{%
4268   \par
4269   \glssubentryitem{##2}%
```

```
4270 \glstarget{##2}{\strut}##4\glspostdescription\space ##6}%
4271 }
```

altlistgroup The altlistgroup glossary style is like the altlist style, but the glossary groups have headings.

```
4272 \newglossarystyle{altlistgroup}{%
```

```
Base it on the altlist style:
```

```
4273 \glossarystyle{altlist}%
```

```
Each group has a heading:
```

```
4274 \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.

```
4275 \newglossarystyle{altlisthypergroup}{%
```

```
Base it on the altlist style:
```

```
4276 \glossarystyle{altlist}%
```

```
Add navigation links at the start of the environment:
```

```
4277 \renewcommand*{\glossaryheader}{%
```

```
4278 \item[\glsnavigation]}%
```

```
Each group has a heading with a hypertext:
```

```
4279 \renewcommand*{\glsgroupheading}[1]{%
```

```
4280 \item[\glsnavhypertext{##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.

```
4281 \newglossarystyle{listdotted}{%
```

```
Base it on the list style:
```

```
4282 \glossarystyle{list}%
```

```
Each main (level 0) entry starts a new item:
```

```
4283 \renewcommand*{\glossaryentryfield}[5]{%
```

```
4284 \item[\makebox[\glslistdottedwidth][l]{%
```

```
4285 \glstarget{##1}{##2}%
```

```
4286 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##3}%
```

```
Sub entries have the same format as main entries:
```

```
4287 \renewcommand*{\glossarysubentryfield}[6]{%
```

```
4288 \item[\makebox[\glslistdottedwidth][l]{%
```

```
4289 \glssubentryitem{##2}%
```

```
4290 \glstarget{##2}{##3}%
```

```
4291 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##4}%
```

```
4292 }
```

`\glslistdottedwidth`

```
4293 \newlength\glslistdottedwidth
4294 \setlength{\glslistdottedwidth}{.5\hsize}
```

`sublistdotted` This style is similar to the `glostylelistdotted` style, except that the main entries just have the name displayed.

```
4295 \newglossarystyle{sublistdotted}{%
```

Base it on the `listdotted` style:

```
4296 \glossarystyle{listdotted}%
```

Main (level 0) entries just display the name:

```
4297 \renewcommand*{\glossaryentryfield}[5]{%
4298   \item[\glsentryitem{##1}\glstarget{##1}{##2}]}%
4299 }
```

3.3 Glossary Styles using `longtable` (the `glossary-long` package)

The glossary styles defined in the package used the `longtable` environment in the glossary.

```
4300 \ProvidesPackage{glossary-long}[2011/03/28 v3.0 (NLCT)]
```

Requires the package:

```
4301 \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`.)

```
4302 \@ifundefined{glsdescwidth}{%
4303   \newlength\glsdescwidth
4304   \setlength{\glsdescwidth}{0.6\hsize}
4305 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column.

```
4306 \@ifundefined{glspagelistwidth}{%
4307   \newlength\glspagelistwidth
4308   \setlength{\glspagelistwidth}{0.1\hsize}
4309 }{}
```

`long` The long glossary style command which uses the `longtable` environment:

```
4310 \newglossarystyle{long}{%
```

Use `longtable` with two columns:

```
4311 \renewenvironment{theglossary}%
4312   {\begin{longtable}[lp{\glsdescwidth}]}%
4313   {\end{longtable}}%
```

Do nothing at the start of the environment:

```
4314 \renewcommand*\glossaryheader}{}
```

No heading between groups:

```
4315 \renewcommand*\glsgroupheading}[1]{}
```

Main (level 0) entries displayed in a row:

```
4316 \renewcommand*\glossaryentryfield}[5]{%
```

```
4317   \glsentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\%
```

Sub entries displayed on the following row without the name:

```
4318 \renewcommand*\glossarysubentryfield}[6]{%
```

```
4319   &
```

```
4320   \glssubentryitem{##2}%
```

```
4321   \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\%
```

Blank row between groups:

```
4322 \renewcommand*\glsgroupskip}{ & \\%
```

```
4323 }
```

longborder The `longborder` style is like the above, but with horizontal and vertical lines:

```
4324 \newglossarystyle{longborder}{%
```

Base it on the `glostylelong` style:

```
4325 \glossarystyle{long}%
```

Use `longtable` with two columns with vertical lines between each column:

```
4326 \renewenvironment{theglossary}{%
```

```
4327   \begin{longtable}{|l|p{\glsdescwidth}|}{\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
4328 \renewcommand*\glossaryheader}{\hline\endhead\hline\endfoot}%
```

```
4329 }
```

longheader The `longheader` style is like the `long` style but with a header:

```
4330 \newglossarystyle{longheader}{%
```

Base it on the `glostylelong` style:

```
4331 \glossarystyle{long}%
```

Set the table's header:

```
4332 \renewcommand*\glossaryheader}{%
```

```
4333   \bfseries \entryname & \bfseries \descriptionname\\endhead}%
```

```
4334 }
```

longheaderborder The `longheaderborder` style is like the `long` style but with a header and border:

```
4335 \newglossarystyle{longheaderborder}{%
```

Base it on the `glostylelongborder` style:

```
4336 \glossarystyle{longborder}%
```

Set the table's header and add horizontal line to table's foot:

```
4337 \renewcommand*\glossaryheader}{%
4338   \hline\bfseries \entryname & \bfseries \descriptionname\\hline
4339   \endhead
4340   \hline\endfoot}%
4341 }
```

`long3col` The `long3col` style is like `long` but with 3 columns

```
4342 \newglossarystyle{long3col}{%
```

Use a `longtable` with 3 columns:

```
4343 \renewenvironment{theglossary}%
4344   {\begin{longtable}[lp{\glsdescwidth}p{\glspagelistwidth}}%
4345   {\end{longtable}}%
```

No table header:

```
4346 \renewcommand*\glossaryheader}{}%
```

No headings between groups:

```
4347 \renewcommand*\glsgroupheading}[1]{%}
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
4348 \renewcommand*\glossaryentryfield}[5]{%
4349   \glstryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\\}%
```

Sub-entries on a separate row (no name, description in second column, page list in third column):

```
4350 \renewcommand*\glossarysubentryfield}[6]{%
4351   &
4352   \glssubentryitem{##2}%
4353   \glstarget{##2}{\strut}##4 & ##6\\}%
```

Blank row between groups:

```
4354 \renewcommand*\glsgroupskip}{ & & \\}%
4355 }
```

`long3colborder` The `long3colborder` style is like the `long3col` style but with a border:

```
4356 \newglossarystyle{long3colborder}{%
```

Base it on the `glostylelong3col` style:

```
4357 \glossarystyle{long3col}%
```

Use a `longtable` with 3 columns with vertical lines around them:

```
4358 \renewenvironment{theglossary}%
4359   {\begin{longtable}[|l|p{\glsdescwidth}|p{\glspagelistwidth}|}%
4360   {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
4361 \renewcommand*\glossaryheader}{\hline\endhead\hline\endfoot}%
4362 }
```

`long3colheader` The `long3colheader` style is like `long3col` but with a header row:

```
4363 \newglossarystyle{long3colheader}{%
      Base it on the glostylelong3col style:
4364 \glossarystyle{long3col}%
      Set the table's header:
4365 \renewcommand*{\glossaryheader}{%
4366   \bfseries\entryname&\bfseries\descriptionname&
4367   \bfseries\pagelistname\\endhead}%
4368 }
```

`long3colheaderborder` The `long3colheaderborder` style is like the above but with a border

```
4369 \newglossarystyle{long3colheaderborder}{%
      Base it on the glostylelong3colborder style:
4370 \glossarystyle{long3colborder}%
      Set the table's header and add horizontal line at table's foot:
4371 \renewcommand*{\glossaryheader}{%
4372   \hline
4373   \bfseries\entryname&\bfseries\descriptionname&
4374   \bfseries\pagelistname\\hline\endhead
4375   \hline\endfoot}%
4376 }
```

`long4col` The `long4col` style has four columns where the third column contains the value of the associated symbol key.

```
4377 \newglossarystyle{long4col}{%
      Use a longtable with 4 columns:
4378 \renewenvironment{theglossary}%
4379   {\begin{longtable}{llll}}%
4380   {\end{longtable}}%
      No table header:
4381 \renewcommand*{\glossaryheader}{}%
      No group headings:
4382 \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):
4383 \renewcommand*{\glossaryentryfield}[5]{%
4384   \glsentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\\}%
      Sub entries on a single row with no name (description in second column, symbol
      in third column, page list in last column):
4385 \renewcommand*{\glossarysubentryfield}[6]{%
4386   &
4387   \glssubentryitem{##2}%
4388   \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%

```

Blank row between groups:

```
4389 \renewcommand*\glsgroupskip}{ & & & \}%  
4390 }
```

long4colheader The `long4colheader` style is like `long4col` but with a header row.

```
4391 \newglossarystyle{long4colheader}{%  
    Base it on the glostylelong4col style:  
4392 \glossarystyle{long4col}%  
    Table has a header:  
4393 \renewcommand*\glossaryheader}{%  
4394     \bfseries\entryname&\bfseries\descriptionname&  
4395     \bfseries \symbolname&  
4396     \bfseries\pagelistname\\\endhead}%  
4397 }
```

long4colborder The `long4colborder` style is like `long4col` but with a border.

```
4398 \newglossarystyle{long4colborder}{%  
    Base it on the glostylelong4col style:  
4399 \glossarystyle{long4col}%  
    Use a longtable with 4 columns surrounded by vertical lines:  
4400 \renewenvironment{theglossary}%  
4401     {\begin{longtable}{|l|l|l|l|}}%  
4402     {\end{longtable}}%  
    Add horizontal lines to the head and foot of the table:  
4403 \renewcommand*\glossaryheader{\hline\endhead\hline\endfoot}%  
4404 }
```

long4colheaderborder The `long4colheaderborder` style is like the above but with a border.

```
4405 \newglossarystyle{long4colheaderborder}{%  
    Base it on the glostylelong4col style:  
4406 \glossarystyle{long4col}%  
    Use a longtable with 4 columns surrounded by vertical lines:  
4407 \renewenvironment{theglossary}%  
4408     {\begin{longtable}{|l|l|l|l|}}%  
4409     {\end{longtable}}%  
    Add table header and horizontal line at the table's foot:  
4410 \renewcommand*\glossaryheader{%  
4411     \hline\bfseries\entryname&\bfseries\descriptionname&  
4412     \bfseries \symbolname&  
4413     \bfseries\pagelistname\\\hline\endhead\hline\endfoot}%  
4414 }
```

altlong4col The `altlong4col` style is like the `long4col` style but can have multiline descriptions and page lists.

```
4415 \newglossarystyle{altlong4col}{%
```

Base it on the `glostylelong4col` style:

```
4416 \glossarystyle{long4col}%  
    Use a longtable with 4 columns where the second and last columns may have  
    multiple lines in each row:  
4417 \renewenvironment{theglossary}%  
4418   {\begin{longtable}{lp{\glsdescwidth}lp{\glspagelistwidth}}}%  
4419   {\end{longtable}}%  
4420 }
```

`altlong4colheader` The `altlong4colheader` style is like `altlong4col` but with a header row.

```
4421 \newglossarystyle{altlong4colheader}{%  
    Base it on the glostylelong4colheader style:  
4422 \glossarystyle{long4colheader}%  
    Use a longtable with 4 columns where the second and last columns may have  
    multiple lines in each row:  
4423 \renewenvironment{theglossary}%  
4424   {\begin{longtable}{lp{\glsdescwidth}lp{\glspagelistwidth}}}%  
4425   {\end{longtable}}%  
4426 }
```

`altlong4colborder` The `altlong4colborder` style is like `altlong4col` but with a border.

```
4427 \newglossarystyle{altlong4colborder}{%  
    Base it on the glostylelong4colborder style:  
4428 \glossarystyle{long4colborder}%  
    Use a longtable with 4 columns where the second and last columns may have  
    multiple lines in each row:  
4429 \renewenvironment{theglossary}%  
4430   {\begin{longtable}{|lp{\glsdescwidth}|lp{\glspagelistwidth}|}}%  
4431   {\end{longtable}}%  
4432 }
```

`altlong4colheaderborder` The `altlong4colheaderborder` style is like the above but with a header as well as a border.

```
4433 \newglossarystyle{altlong4colheaderborder}{%  
    Base it on the glostylelong4colheaderborder style:  
4434 \glossarystyle{long4colheaderborder}%  
    Use a longtable with 4 columns where the second and last columns may have  
    multiple lines in each row:  
4435 \renewenvironment{theglossary}%  
4436   {\begin{longtable}{|lp{\glsdescwidth}|lp{\glspagelistwidth}|}}%  
4437   {\end{longtable}}%  
4438 }
```

3.4 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.

4439 \ProvidesPackage{glossary-longragged}[2011/03/28 v3.0 (NLCT)]

Requires the package:

4440 \RequirePackage{array}

Requires the package:

4441 \RequirePackage{longtable}

`\glsdescwidth` This is a length that governs the width of the description column. This may have already been defined.

```
4442 \@ifundefined{glsdescwidth}{%
4443   \newlength{glsdescwidth
4444   \setlength{glsdescwidth}{0.6\hsize}
4445 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

```
4446 \@ifundefined{glspagelistwidth}{%
4447   \newlength{glspagelistwidth
4448   \setlength{glspagelistwidth}{0.1\hsize}
4449 }{}
```

`longragged` The longragged glossary style is like the long but uses ragged right formatting for the description column.

4450 \newglossarystyle{longragged}{%

Use longtable with two columns:

```
4451 \renewenvironment{theglossary}%
4452   {\begin{longtable}[1>{\raggedright}p{glsdescwidth}}}%
4453   {\end{longtable}}%
```

Do nothing at the start of the environment:

4454 \renewcommand*{\glossaryheader}{}%

No heading between groups:

4455 \renewcommand*{\glsgroupheading}[1]{}%

Main (level 0) entries displayed in a row:

```
4456 \renewcommand*{\glossaryentryfield}[5]{%
4457   \glstentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%
4458   \tabularnewline}%
```

Sub entries displayed on the following row without the name:

```
4459 \renewcommand*{\glossarysubentryfield}[6]{%
4460   &
4461   \glssubentryitem{##2}%
4462   \glstarget{##2}{\strut}##4\glspostdescription\space ##6%
4463   \tabularnewline}%
```

Blank row between groups:

```
4464 \renewcommand*\glsgroupskip}{ & \tabularnewline}%  
4465 }
```

`longraggedborder` The `longraggedborder` style is like the above, but with horizontal and vertical lines:

```
4466 \newglossarystyle{longraggedborder}{%
```

Base it on the `glostylelongragged` style:

```
4467 \glossarystyle{longragged}%
```

Use `longtable` with two columns with vertical lines between each column:

```
4468 \renewenvironment{theglossary}{%
```

```
4469 \begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|}%
```

```
4470 {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
4471 \renewcommand*\glossaryheader}{\hline\endhead\hline\endfoot}%
```

```
4472 }
```

`longraggedheader` The `longraggedheader` style is like the `longragged` style but with a header:

```
4473 \newglossarystyle{longraggedheader}{%
```

Base it on the `glostylelongragged` style:

```
4474 \glossarystyle{longragged}%
```

Set the table's header:

```
4475 \renewcommand*\glossaryheader}{%
```

```
4476 \bfseries \entryname & \bfseries \descriptionname
```

```
4477 \tabularnewline\endhead}%
```

```
4478 }
```

`longraggedheaderborder` The `longraggedheaderborder` style is like the `longragged` style but with a header and border:

```
4479 \newglossarystyle{longraggedheaderborder}{%
```

Base it on the `glostylelongraggedborder` style:

```
4480 \glossarystyle{longraggedborder}%
```

Set the table's header and add horizontal line to table's foot:

```
4481 \renewcommand*\glossaryheader}{%
```

```
4482 \hline\bfseries \entryname & \bfseries \descriptionname
```

```
4483 \tabularnewline\hline
```

```
4484 \endhead
```

```
4485 \hline\endfoot}%
```

```
4486 }
```

`longragged3col` The `longragged3col` style is like `longragged` but with 3 columns

```
4487 \newglossarystyle{longragged3col}{%
```

Use a `longtable` with 3 columns:

```
4488 \renewenvironment{theglossary}%  
4489   {\begin{longtable}[1]>{\raggedright}p{\glsdescwidth}%  
4490     >{\raggedright}p{\glspagelistwidth}}}%  
4491   {\end{longtable}}%
```

No table header:

```
4492 \renewcommand*{\glossaryheader}{}%
```

No headings between groups:

```
4493 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
4494 \renewcommand*{\glossaryentryfield}[5]{%  
4495   \glstarget{##1}\glstarget{##1}{##2} & ##3 & ##5\tabularnewline}%
```

Sub-entries on a separate row (no name, description in second column, page list in third column):

```
4496 \renewcommand*{\glossarysubentryfield}[6]{%  
4497   &  
4498   \glssubentryitem{##2}%  
4499   \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%
```

Blank row between groups:

```
4500 \renewcommand*{\glsgroupskip}{ & \tabularnewline}%  
4501 }
```

`longragged3colborder` The `longragged3colborder` style is like the `longragged3col` style but with a border:

```
4502 \newglossarystyle{longragged3colborder}{%
```

Base it on the `glostylelongragged3col` style:

```
4503 \glossarystyle{longragged3col}%
```

Use a `longtable` with 3 columns with vertical lines around them:

```
4504 \renewenvironment{theglossary}%  
4505   {\begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|%  
4506     >{\raggedright}p{\glspagelistwidth}|}}}%  
4507   {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
4508 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%  
4509 }
```

`longragged3colheader` The `longragged3colheader` style is like `longragged3col` but with a header row:

```
4510 \newglossarystyle{longragged3colheader}{%
```

Base it on the `glostylelongragged3col` style:

```
4511 \glossarystyle{longragged3col}%
```

Set the table's header:

```
4512 \renewcommand*\glossaryheader}{%
4513   \bfseries\entryname&\bfseries\descriptionname&
4514   \bfseries\pagelistname\tabularnewline\endhead}%
4515 }
```

longragged3colheaderborder The longragged3colheaderborder style is like the above but with a border

```
4516 \newglossarystyle{longragged3colheaderborder}{%
```

Base it on the glostylelongragged3colborder style:

```
4517 \glossarystyle{longragged3colborder}{%
```

Set the table's header and add horizontal line at table's foot:

```
4518 \renewcommand*\glossaryheader}{%
4519   \hline
4520   \bfseries\entryname&\bfseries\descriptionname&
4521   \bfseries\pagelistname\tabularnewline\hline\endhead
4522   \hline\endfoot}%
4523 }
```

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.

```
4524 \newglossarystyle{altlongragged4col}{%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
4525 \renewenvironment{theglossary}{%
4526   {\begin{longtable}[1]>{\raggedright}p{\glsdescwidth}l%
4527     >{\raggedright}p{\glspagelistwidth}}}%
4528   {\end{longtable}}%
```

No table header:

```
4529 \renewcommand*\glossaryheader}{}
```

No group headings:

```
4530 \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):

```
4531 \renewcommand*\glossaryentryfield}[5]{%
4532   \glsentryitem{##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):

```
4533 \renewcommand*\glossarysubentryfield}[6]{%
4534   &
4535   \glssubentryitem{##2}%
4536   \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%
```

Blank row between groups:

```
4537 \renewcommand*\glsgroupskip}{ & & \tabularnewline}%
4538 }
```

`altlongragged4colheader` The `altlongragged4colheader` style is like `altlongragged4col` but with a header row.

```
4539 \newglossarystyle{altlongragged4colheader}{%
```

Base it on the `glostylealtlongragged4col` style:

```
4540 \glossarystyle{altlongragged4col}%
```

Use a `longtable` with 4 columns where the second and last columns may have multiple lines in each row:

```
4541 \renewenvironment{theglossary}%
```

```
4542   {\begin{longtable}{1>{\raggedright}p{\glsdescwidth}1%
```

```
4543     >{\raggedright}p{\glspagelistwidth}}%
```

```
4544   {\end{longtable}}%
```

Table has a header:

```
4545 \renewcommand*{\glossaryheader}{%
```

```
4546   \bfseries\entryname&\bfseries\descriptionname&
```

```
4547   \bfseries \symbolname&
```

```
4548   \bfseries\pagelistname\tabularnewline\endhead}%
```

```
4549 }
```

`altlongragged4colborder` The `altlongragged4colborder` style is like `altlongragged4col` but with a border.

```
4550 \newglossarystyle{altlongragged4colborder}{%
```

Base it on the `glostylealtlongragged4col` style:

```
4551 \glossarystyle{altlongragged4col}%
```

Use a `longtable` with 4 columns where the second and last columns may have multiple lines in each row:

```
4552 \renewenvironment{theglossary}%
```

```
4553   {\begin{longtable}{|1|>{\raggedright}p{\glsdescwidth}|1|%
```

```
4554     >{\raggedright}p{\glspagelistwidth}|}}%
```

```
4555   {\end{longtable}}%
```

Add horizontal lines to the head and foot of the table:

```
4556 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
```

```
4557 }
```

`altlongragged4colheaderborder` The `altlongragged4colheaderborder` style is like the above but with a header as well as a border.

```
4558 \newglossarystyle{altlongragged4colheaderborder}{%
```

Base it on the `glostylealtlongragged4col` style:

```
4559 \glossarystyle{altlongragged4col}%
```

Use a `longtable` with 4 columns where the second and last columns may have multiple lines in each row:

```
4560 \renewenvironment{theglossary}%
```

```
4561   {\begin{longtable}{|1|>{\raggedright}p{\glsdescwidth}|1|%
```

```
4562     >{\raggedright}p{\glspagelistwidth}|}}%
```

```
4563   {\end{longtable}}%
```

Add table header and horizontal line at the table's foot:

```
4564 \renewcommand*{\glossaryheader}{%
4565   \hline\bfseries\entryname&\bfseries\descriptionname&
4566   \bfseries \symbolname&
4567   \bfseries\pagelistname\tabularnewline\hline\endhead
4568   \hline\endfoot}%
4569 }
```

3.5 Glossary Styles using supertabular environment (glossary-super package)

The glossary styles defined in the package use the supertabular environment.

```
4570 \ProvidesPackage{glossary-super}[2011/03/28 v3.0 (NLCT)]
```

Requires the package:

```
4571 \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.

```
4572 \@ifundefined{glsdescwidth}{%
4573   \newlength\glsdescwidth
4574   \setlength{\glsdescwidth}{0.6\hsize}
4575 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined if has been loaded.

```
4576 \@ifundefined{glspagelistwidth}{%
4577   \newlength\glspagelistwidth
4578   \setlength{\glspagelistwidth}{0.1\hsize}
4579 }{}
```

`super` The super glossary style uses the supertabular environment (it uses lengths defined in the package.)

```
4580 \newglossarystyle{super}{%
```

Put the glossary in a supertabular environment with two columns and no head or tail:

```
4581 \renewenvironment{theglossary}%
4582   {\tablehead{} \tabletail{}%
4583   \begin{supertabular}{lp{\glsdescwidth}}%
4584   {\end{supertabular}}%
```

Do nothing at the start of the table:

```
4585 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
4586 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries put in a row (name in first column, description and page list in second column):

```
4587 \renewcommand*\glossaryentryfield}[5]{%
4588   \glstryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\}%
```

Sub entries put in a row (no name, description and page list in second column):

```
4589 \renewcommand*\glossarysubentryfield}[6]{%
4590   &
4591   \glssubentryitem{##2}%
4592   \glstarget{##2}{\strut}##4\glspostdescription\space ##6\}%
```

Blank row between groups:

```
4593 \renewcommand*\glsgroupskip}{ & \}%
4594 }
```

superborder The superborder style is like the above, but with horizontal and vertical lines:

```
4595 \newglossarystyle{superborder}{%
```

Base it on the glostylesuper style:

```
4596 \glossarystyle{super}%
```

Put the glossary in a supertabular environment with two columns and a horizontal line in the head and tail:

```
4597 \renewenvironment{theglossary}%
4598   {\tablehead{\hline}\tabletail{\hline}%
4599   \begin{supertabular}{|l|p{\glstdescwidth}|}%
4600   {\end{supertabular}}%
4601 }
```

superheader The superheader style is like the super style, but with a header:

```
4602 \newglossarystyle{superheader}{%
```

Base it on the glostylesuper style:

```
4603 \glossarystyle{super}%
```

Put the glossary in a supertabular environment with two columns, a header and no tail:

```
4604 \renewenvironment{theglossary}%
4605   {\tablehead{\bfseries \entryname & \bfseries \descriptionname\}%
4606   \tabletail{}}%
4607   \begin{supertabular}{lp{\glstdescwidth}}%
4608   {\end{supertabular}}%
4609 }
```

superheaderborder The superheaderborder style is like the super style but with a header and border:

```
4610 \newglossarystyle{superheaderborder}{%
```

Base it on the glostylesuper style:

```
4611 \glossarystyle{super}%
```

Put the glossary in a supertabular environment with two columns, a header and horizontal lines above and below the table:

```

4612 \renewenvironment{theglossary}%
4613   {\tablehead{\hline\bfseries \entryname &
4614             \bfseries \descriptionname\\ \hline}%
4615     \tabletail{\hline}
4616     \begin{supertabular}{|l|p{\glsdescwidth}|}%
4617     {\end{supertabular}}%
4618 }

```

super3col The super3col style is like the super style, but with 3 columns:

```

4619 \newglossarystyle{super3col}{%

```

Put the glossary in a supertabular environment with three columns and no head or tail:

```

4620 \renewenvironment{theglossary}%
4621   {\tablehead{}\tabletail{}}%
4622   \begin{supertabular}{lp{\glsdescwidth}p{\glspagelistwidth}}%
4623   {\end{supertabular}}%

```

Do nothing at the start of the table:

```

4624 \renewcommand*{\glossaryheader}{}%

```

No group headings:

```

4625 \renewcommand*{\glsgroupheading}[1]{}%

```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```

4626 \renewcommand*{\glossaryentryfield}[5]{%
4627   \glstarget{##1}\glstarget{##1}{##2} & ##3 & ##5\}%

```

Sub entries on a row (no name, description in second column, page list in last column):

```

4628 \renewcommand*{\glossarysubentryfield}[6]{%
4629   &
4630   \glssubentryitem{##2}%
4631   \glstarget{##2}{\strut}##4 & ##6\}%

```

Blank row between groups:

```

4632 \renewcommand*{\glsgroupskip}{ & \}%
4633 }

```

super3colborder The super3colborder style is like the super3col style, but with a border:

```

4634 \newglossarystyle{super3colborder}{%

```

Base it on the glostylesuper3col style:

```

4635 \glossarystyle{super3col}%

```

Put the glossary in a supertabular environment with three columns and a horizontal line in the head and tail:

```

4636 \renewenvironment{theglossary}%
4637   {\tablehead{\hline}\tabletail{\hline}%

```

```

4638     \begin{supertabular}{|l|p{\glsdescwidth}|p{\glspagelistwidth}|}%
4639     {\end{supertabular}}%
4640 }

```

super3colheader The `super3colheader` style is like the `super3col` style but with a header row:

```

4641 \newglossarystyle{super3colheader}{%
    Base it on the glostylesuper3col style:
4642   \glossarystyle{super3col}%
    Put the glossary in a supertabular environment with three columns, a header and
    no tail:
4643   \renewenvironment{theglossary}%
4644     {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
4645               \bfseries\pagelistname\\}\tabletail{}}%
4646     \begin{supertabular}{|p{\glsdescwidth}p{\glspagelistwidth}|}%
4647     {\end{supertabular}}%
4648 }

```

super3colheaderborder The `super3colheaderborder` style is like the `super3col` style but with a header and border:

```

4649 \newglossarystyle{super3colheaderborder}{%
    Base it on the glostylesuper3colborder style:
4650   \glossarystyle{super3colborder}%
    Put the glossary in a supertabular environment with three columns, a header with
    horizontal lines and a horizontal line in the tail:
4651   \renewenvironment{theglossary}%
4652     {\tablehead{\hline
4653               \bfseries\entryname&\bfseries\descriptionname&
4654               \bfseries\pagelistname\\\hline}%
4655     \tabletail{\hline}%
4656     \begin{supertabular}{|l|p{\glsdescwidth}|p{\glspagelistwidth}|}%
4657     {\end{supertabular}}%
4658 }

```

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.

```

4659 \newglossarystyle{super4col}{%
    Put the glossary in a supertabular environment with four columns and no head or
    tail:
4660   \renewenvironment{theglossary}%
4661     {\tablehead{} \tabletail{}}%
4662     \begin{supertabular}{|l|l|l|l|}%
4663     \end{supertabular}}%
    Do nothing at the start of the table:
4664   \renewcommand*{\glossaryheader}{}%

```

No group headings:

```
4665 \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:

```
4666 \renewcommand*\glossaryentryfield}[5]{%
4667   \glsentryitem{##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:

```
4668 \renewcommand*\glossarysubentryfield}[6]{%
4669   &
4670   \glssubentryitem{##2}%
4671   \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%
```

Blank row between groups:

```
4672 \renewcommand*\glsgroupskip}{ & & & \\}%
4673 }
```

super4colheader The `super4colheader` style is like the `super4col` but with a header row.

```
4674 \newglossarystyle{super4colheader}{%
```

Base it on the `glostylesuper4col` style:

```
4675 \glossarystyle{super4col}%
```

Put the glossary in a `supertabular` environment with four columns, a header and no tail:

```
4676 \renewenvironment{theglossary}%
4677   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
4678     \bfseries\symbolname &
4679     \bfseries\pagelistname\\}%
4680   \tabletail{}}%
4681   \begin{supertabular}{|l|l|l|l|}%
4682   {\end{supertabular}}%
4683 }
```

super4colborder The `super4colborder` style is like the `super4col` but with a border.

```
4684 \newglossarystyle{super4colborder}{%
```

Base it on the `glostylesuper4col` style:

```
4685 \glossarystyle{super4col}%
```

Put the glossary in a `supertabular` environment with four columns and a horizontal line in the head and tail:

```
4686 \renewenvironment{theglossary}%
4687   {\tablehead{\hline}\tabletail{\hline}%
4688   \begin{supertabular}{|l|l|l|l|}%
4689   {\end{supertabular}}%
4690 }
```

super4colheaderborder The `super4colheaderborder` style is like the `super4col` but with a header and border.

```
4691 \newglossarystyle{super4colheaderborder}{%
```

Base it on the `glostylesuper4col` style:

```
4692 \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:  
4693 \renewenvironment{theglossary}%  
4694   {\tablehead{\hline\bfseries\entryname&\bfseries\descriptionname&  
4695     \bfseries\symbolname &  
4696     \bfseries\pagelistname\\\hline}\tabletail{\hline}%  
4697   \begin{supertabular}{|1|1|1|1|}%  
4698   {\end{supertabular}}%  
4699 }
```

`altsuper4col` The `altsuper4col` glossary style is like `super4col` but has provision for multiline descriptions.

```
4700 \newglossarystyle{altsuper4col}{%  
Base it on the glostylesuper4col style:  
4701 \glossarystyle{super4col}%  
Put the glossary in a supertabular environment with four columns and no head or  
tail:  
4702 \renewenvironment{theglossary}%  
4703   {\tablehead{}\tabletail{}%  
4704   \begin{supertabular}{lp{\glsdescwidth}lp{\glspagelistwidth}}%  
4705   {\end{supertabular}}%  
4706 }
```

`altsuper4colheader` The `altsuper4colheader` style is like the `altsuper4col` but with a header row.

```
4707 \newglossarystyle{altsuper4colheader}{%  
Base it on the glostylesuper4colheader style:  
4708 \glossarystyle{super4colheader}%  
Put the glossary in a supertabular environment with four columns, a header and  
no tail:  
4709 \renewenvironment{theglossary}%  
4710   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&  
4711     \bfseries\symbolname &  
4712     \bfseries\pagelistname\\\tabletail{}%  
4713   \begin{supertabular}{lp{\glsdescwidth}lp{\glspagelistwidth}}%  
4714   {\end{supertabular}}%  
4715 }
```

`altsuper4colborder` The `altsuper4colborder` style is like the `altsuper4col` but with a border.

```
4716 \newglossarystyle{altsuper4colborder}{%  
Base it on the glostylesuper4colborder style:  
4717 \glossarystyle{super4colborder}%
```

Put the glossary in a supertabular environment with four columns and a horizontal line in the head and tail:

```
4718 \renewenvironment{theglossary}%
4719   {\tablehead{\hline}\tabletail{\hline}%
4720    \begin{supertabular}%
4721     {|l|p{\glsdescwidth}|l|p{\glspagelistwidth}|}}%
4722   {\end{supertabular}}%
4723 }
```

`altsuper4colheaderborder` The `altsuper4colheaderborder` style is like the `altsuper4col` but with a header and border.

```
4724 \newglossarystyle{altsuper4colheaderborder}{%
```

Base it on the `glostylesuper4colheaderborder` style:

```
4725 \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:

```
4726 \renewenvironment{theglossary}%
4727   {\tablehead{\hline
4728    \bfseries\entryname &
4729    \bfseries\descriptionname &
4730    \bfseries\symbolname &
4731    \bfseries\pagelistname\\ \hline}%
4732   \tabletail{\hline}%
4733   \begin{supertabular}%
4734    {|l|p{\glsdescwidth}|l|p{\glspagelistwidth}|}}%
4735   {\end{supertabular}}%
4736 }
```

3.6 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.

```
4737 \ProvidesPackage{glossary-superragged}[2011/03/28 v3.0 (NLCT)]
```

Requires the package:

```
4738 \RequirePackage{array}
```

Requires the package:

```
4739 \RequirePackage{supertabular}
```

`\glsdescwidth` This is a length that governs the width of the description column. This may already have been defined.

```
4740 \@ifundefined{glsdescwidth}{%
4741   \newlength\glsdescwidth
4742   \setlength{\glsdescwidth}{0.6\hsize}
4743 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

```
4744 \ifundefined{glspagelistwidth}{%
4745   \newlength{glspagelistwidth}
4746   \setlength{glspagelistwidth}{0.1\hsize}
4747 }
```

`superragged` The `superragged` glossary style uses the `supertabular` environment.

```
4748 \newglossarystyle{superragged}{%
```

Put the glossary in a `supertabular` environment with two columns and no head or tail:

```
4749   \renewenvironment{theglossary}%
4750     {\tablehead{ }\tabletail{ }}%
4751     \begin{supertabular}{|l>{\raggedright}p{\glstdescwidth}|}%
4752     {\end{supertabular}}%
```

Do nothing at the start of the table:

```
4753   \renewcommand*{\glossaryheader}{ }%
```

No group headings:

```
4754   \renewcommand*{\glsgroupheading}[1]{ }
```

Main (level 0) entries put in a row (name in first column, description and page list in second column):

```
4755   \renewcommand*{\glossaryentryfield}[5]{%
4756     \glstentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%
4757     \tabularnewline}%
```

Sub entries put in a row (no name, description and page list in second column):

```
4758   \renewcommand*{\glossarysubentryfield}[6]{%
4759     &
4760     \glssubentryitem{##2}%
4761     \glstarget{##2}{\strut}##4\glspostdescription\space ##6%
4762     \tabularnewline}%
```

Blank row between groups:

```
4763   \renewcommand*{\glsgroupskip}{ & \tabularnewline}%
4764 }
```

`superraggedborder` The `superraggedborder` style is like the above, but with horizontal and vertical lines:

```
4765 \newglossarystyle{superraggedborder}{%
```

Base it on the `glostylesuperragged` style:

```
4766   \glossarystyle{superragged}%
```

Put the glossary in a `supertabular` environment with two columns and a horizontal line in the head and tail:

```
4767   \renewenvironment{theglossary}%
4768     {\tablehead{\hline}\tabletail{\hline}%
4769     \begin{supertabular}{|l|>{\raggedright}p{\glstdescwidth}|}%
```

```
4770   {\end{supertabular}}%
4771 }
```

superraggedheader The superraggedheader style is like the super style, but with a header:

```
4772 \newglossarystyle{superraggedheader}{%
  Base it on the glostylesuperragged style:
4773   \glossarystyle{superragged}%
  Put the glossary in a supertabular environment with two columns, a header and
  no tail:
4774   \renewenvironment{theglossary}%
4775     {\tablehead{\bfseries \entryname & \bfseries \descriptionname
4776       \tabularnewline}%
4777     \tabletail{}}%
4778     \begin{supertabular}{1>{\raggedright}p{\glsdescwidth}}%
4779     {\end{supertabular}}}%
4780 }
```

superraggedheaderborder The superraggedheaderborder style is like the superragged style but with a header and border:

```
4781 \newglossarystyle{superraggedheaderborder}{%
  Base it on the glostylesuper style:
4782   \glossarystyle{superragged}%
  Put the glossary in a supertabular environment with two columns, a header and
  horizontal lines above and below the table:
4783   \renewenvironment{theglossary}%
4784     {\tablehead{\hline\bfseries \entryname &
4785       \bfseries \descriptionname\tabularnewline\hline}%
4786     \tabletail{\hline}}%
4787     \begin{supertabular}{|1>{\raggedright}p{\glsdescwidth}|}%
4788     {\end{supertabular}}}%
4789 }
```

superragged3col The superragged3col style is like the superragged style, but with 3 columns:

```
4790 \newglossarystyle{superragged3col}{%
  Put the glossary in a supertabular environment with three columns and no head
  or tail:
4791   \renewenvironment{theglossary}%
4792     {\tablehead{}\tabletail{}}%
4793     \begin{supertabular}{1>{\raggedright}p{\glsdescwidth}%
4794       >{\raggedright}p{\glspagelistwidth}}%
4795     {\end{supertabular}}%
  Do nothing at the start of the table:
4796   \renewcommand*{\glossaryheader}{}%
  No group headings:
4797   \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
4798 \renewcommand*\glossaryentryfield}[5]{%
4799   \glstentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\tabularnewline}%
```

Sub entries on a row (no name, description in second column, page list in last column):

```
4800 \renewcommand*\glossarysubentryfield}[6]{%
4801   &
4802   \glssubentryitem{##2}%
4803   \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%
```

Blank row between groups:

```
4804 \renewcommand*\glsgroupskip}{ & \tabularnewline}%
4805 }
```

superragged3colborder The `superragged3colborder` style is like the `superragged3col` style, but with a border:

```
4806 \newglossarystyle{superragged3colborder}{%
  Base it on the glostylesuperragged3col style:
4807 \glossarystyle{superragged3col}%
  Put the glossary in a supertabular environment with three columns and a horizontal
  line in the head and tail:
4808 \renewenvironment{theglossary}%
4809   {\tablehead{\hline}\tabletail{\hline}%
4810    \begin{supertabular}{|l|>{\raggedright}p{\glsdescwidth}|%
4811     >{\raggedright}p{\glspagerlistwidth}|}}%
4812   {\end{supertabular}}%
4813 }
```

superragged3colheader The `superragged3colheader` style is like the `superragged3col` style but with a header row:

```
4814 \newglossarystyle{superragged3colheader}{%
  Base it on the glostylesuperragged3col style:
4815 \glossarystyle{superragged3col}%
  Put the glossary in a supertabular environment with three columns, a header and
  no tail:
4816 \renewenvironment{theglossary}%
4817   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
4818    \bfseries\pagerlistname\tabularnewline}\tabletail{}}%
4819   \begin{supertabular}{l>{\raggedright}p{\glsdescwidth}%
4820    >{\raggedright}p{\glspagerlistwidth}}}%
4821   {\end{supertabular}}%
4822 }
```

superraggedright3colheaderborder The `superragged3colheaderborder` style is like the `superragged3col` style but with a header and border:

```
4823 \newglossarystyle{superragged3colheaderborder}{%
```

Base it on the `glostylesuperragged3colborder` style:

```
4824 \glossarystyle{superragged3colborder}%
Put the glossary in a supertabular environment with three columns, a header with
horizontal lines and a horizontal line in the tail:
4825 \renewenvironment{theglossary}%
4826   {\tablehead{\hline
4827     \bfseries\entryname&\bfseries\descriptionname&
4828     \bfseries\pagelistname\tabularnewline\hline}%
4829   \tabletail{\hline}%
4830   \begin{supertabular}{|l|>{\raggedright}p{\glsdescwidth}|%
4831     >{\raggedright}p{\glspagelistwidth}|}%
4832   {\end{supertabular}}%
4833 }
```

`altsuperragged4col` The `altsuperragged4col` glossary style is like `altsuper4col` style in the package but uses ragged right formatting in the description and page list columns.

```
4834 \newglossarystyle{altsuperragged4col}{%
Put the glossary in a supertabular environment with four columns and no head or
tail:
4835 \renewenvironment{theglossary}%
4836   {\tablehead{}\tabletail{}%
4837   \begin{supertabular}{|l>{\raggedright}p{\glsdescwidth}l%
4838     >{\raggedright}p{\glspagelistwidth}|}%
4839   {\end{supertabular}}%
```

Do nothing at the start of the table:

```
4840 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
4841 \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:

```
4842 \renewcommand*{\glossaryentryfield}[5]{%
4843   \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:

```
4844 \renewcommand*{\glossarysubentryfield}[6]{%
4845   &
4846   \glssubentryitem{##2}%
4847   \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%
```

Blank row between groups:

```
4848 \renewcommand*{\glsgroupskip}{ & & \tabularnewline}%
4849 }
```

`altsuperragged4colheader` The `altsuperragged4colheader` style is like the `altsuperragged4col` style but with a header row.

```
4850 \newglossarystyle{altsuperragged4colheader}{%
```

Base it on the `glostylealtsuperragged4col` style:

```
4851 \glossarystyle{altsuperragged4col}%
Put the glossary in a supertabular environment with four columns, a header and
no tail:
4852 \renewenvironment{theglossary}%
4853   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
4854     \bfseries\symbolname &
4855     \bfseries\pagelistname\tabularnewline}\tabletail{}}%
4856   \begin{supertabular}{l>{\raggedright}p{\glsdescwidth}l%
4857     >{\raggedright}p{\glspagelistwidth}}}%
4858   {\end{supertabular}}}%
4859 }
```

`altsuperragged4colborder` The `altsuperragged4colborder` style is like the `altsuperragged4col` style but with a border.

```
4860 \newglossarystyle{altsuperragged4colborder}{%
Base it on the glostylealtsuperragged4col style:
4861 \glossarystyle{altsuper4col}%
Put the glossary in a supertabular environment with four columns and a horizontal
line in the head and tail:
4862 \renewenvironment{theglossary}%
4863   {\tablehead{\hline}\tabletail{\hline}%
4864   \begin{supertabular}%
4865     {l|>{\raggedright}p{\glsdescwidth}|l|%
4866     >{\raggedright}p{\glspagelistwidth}|}}%
4867   {\end{supertabular}}}%
4868 }
```

`altsuperragged4colheaderborder` The `altsuperragged4colheaderborder` style is like the `altsuperragged4col` style but with a header and border.

```
4869 \newglossarystyle{altsuperragged4colheaderborder}{%
Base it on the glostylealtsuperragged4col style:
4870 \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:
4871 \renewenvironment{theglossary}%
4872   {\tablehead{\hline
4873     \bfseries\entryname &
4874     \bfseries\descriptionname &
4875     \bfseries\symbolname &
4876     \bfseries\pagelistname\tabularnewline\hline}%
4877   \tabletail{\hline}%
4878   \begin{supertabular}%
4879     {l|>{\raggedright}p{\glsdescwidth}|l|%
4880     >{\raggedright}p{\glspagelistwidth}|}}%
4881   {\end{supertabular}}}%
4882 }
```

3.7 Tree Styles (glossary-tree.sty)

The style file defines glossary styles that have a tree-like structure. These are designed for hierarchical glossaries.

```
4883 \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.

```
4884 \newglossarystyle{index}{%
```

Set the paragraph indentation and skip and define `\item` to be the same as that used by `theindex`:

```
4885 \renewenvironment{theglossary}%  
4886   {\setlength{\parindent}{0pt}}%  
4887   \setlength{\parskip}{0pt plus 0.3pt}}%  
4888   \let\item\@idxitem}%  
4889   {}%
```

Do nothing at the start of the environment:

```
4890 \renewcommand*{\glossaryheader}{}%
```

No group headers:

```
4891 \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.

```
4892 \renewcommand*{\glossaryentryfield}[5]{}%  
4893 \item\glsentryitem{##1}\textbf{\glstarget{##1}{##2}}%  
4894 \ifx\relax##4\relax  
4895 \else  
4896   \space{##4}%  
4897 \fi  
4898 \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.

```
4899 \renewcommand*{\glossarysubentryfield}[6]{}%  
4900   \ifcase##1\relax  
4901     % level 0  
4902     \item  
4903   \or  
4904     % level 1  
4905     \subitem  
4906     \glssubentryitem{##2}%  
4907   \else  
4908     % all other levels  
4909     \subsubitem
```

```

4910 \fi
4911 \textbf{\glstarget{##2}{##3}}%
4912 \ifx\relax##5\relax
4913 \else
4914 \space{##5}%
4915 \fi
4916 \space##4\glspostdescription\space ##6}%

```

Vertical gap between groups is the same as that used by indices:

```

4917 \renewcommand*\glsgroupskip}{\indexspace}}

```

indexgroup The `indexgroup` style is like the `index` style but has headings.

```

4918 \newglossarystyle{indexgroup}{%

```

Base it on the `glostyleindex` style:

```

4919 \glossarystyle{index}%

```

Add a heading for each group. This puts the group's title in bold followed by a vertical gap.

```

4920 \renewcommand*\glsgroupheading}[1]{%
4921 \item\textbf{\glsgrouptitle{##1}}\indexspace}%
4922 }

```

indexhypergroup The `indexhypergroup` style is like the `indexgroup` style but has hyper navigation.

```

4923 \newglossarystyle{indexhypergroup}{%

```

Base it on the `glostyleindex` style:

```

4924 \glossarystyle{index}%

```

Put navigation links to the groups at the start of the glossary:

```

4925 \renewcommand*\glossaryheader}{%
4926 \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.

```

4927 \renewcommand*\glsgroupheading}[1]{%
4928 \item\textbf{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
4929 \indexspace}%
4930 }

```

tree The `tree` glossary style is similar in style to the `index` style, but can have arbitrary levels.

```

4931 \newglossarystyle{tree}{%

```

Set the paragraph indentation and skip:

```

4932 \renewenvironment{theglossary}%
4933 {\setlength{\parindent}{0pt}%
4934 \setlength{\parskip}{0pt plus 0.3pt}}%
4935 {}%

```

Do nothing at the start of the `theglossary` environment:

```

4936 \renewcommand*\glossaryheader}{}%

```

No group headings:

```
4937 \renewcommand*\glsgroupheading}[1]{%
```

Main (level 0) entries: name in bold, followed by symbol in brackets (if it exists), the description and the page list:

```
4938 \renewcommand{\glossaryentryfield}[5]{%
4939   \hangindent0pt\relax
4940   \parindent0pt\relax
4941   \glentryitem{##1}\textbf{\glstarget{##1}{##2}}%
4942   \ifx\relax##4\relax
4943   \else
4944     \space{##4}%
4945   \fi
4946   \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.

```
4947 \renewcommand{\glossarysubentryfield}[6]{%
4948   \hangindent##1\glstreeindent\relax
4949   \parindent##1\glstreeindent\relax
4950   \ifnum##1=1\relax
4951     \glssubentryitem{##2}%
4952   \fi
4953   \textbf{\glstarget{##2}{##3}}%
4954   \ifx\relax##5\relax
4955   \else
4956     \space{##5}%
4957   \fi
4958   \space##4\glspostdescription\space ##6\par}%
```

Vertical gap between groups is the same as that used by indices:

```
4959 \renewcommand*\glsgroupskip{\indexspace}}
```

treegroup Like the tree style but the glossary groups have headings.

```
4960 \newglossarystyle{treegroup}{%
```

Base it on the `glostyletree` style:

```
4961 \glossarystyle{tree}%
```

Each group has a heading (in bold) followed by a vertical gap):

```
4962 \renewcommand{\glsgroupheading}[1]{\par
4963   \noindent\textbf{\glsetgrouptitle{##1}}\par\indexspace}%
4964 }
```

treehypergroup The `treehypergroup` style is like the `treegroup` style, but has a set of links to the groups at the start of the glossary.

```
4965 \newglossarystyle{treehypergroup}{%
```

Base it on the `glostyletree` style:

```
4966 \glossarystyle{tree}%
```

Put navigation links to the groups at the start of the theglossary environment:

```
4967 \renewcommand*\glossaryheader}{%
4968   \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
4969 \renewcommand*\glsgroupheading}[1]{%
4970   \par\noindent
4971   \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
4972   \indexspace}%
4973 }
```

`\glstreeindent` Length governing left indent for each level of the tree style.

```
4974 \newlength\glstreeindent
4975 \setlength{\glstreeindent}{10pt}
```

`treenoname` The `treenoname` glossary style is like the tree style, but doesn't print the name or symbol for sub-levels.

```
4976 \newglossarystyle{treenoname}{%
```

Set the paragraph indentation and skip:

```
4977 \renewenvironment{theglossary}%
4978   {\setlength{\parindent}{0pt}%
4979   \setlength{\parskip}{0pt plus 0.3pt}}%
4980   {}%
```

No header:

```
4981 \renewcommand*\glossaryheader}{}%
```

No group headings:

```
4982 \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.

```
4983 \renewcommand{\glossaryentryfield}[5]{%
4984   \hangindent0pt\relax
4985   \parindent0pt\relax
4986   \glsentryitem{##1}\textbf{\glsstarget{##1}{##2}}%
4987   \ifx\relax##4\relax
4988     \else
4989     \space{##4}%
4990     \fi
4991   \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.

```
4992 \renewcommand{\glossarysubentryfield}[6]{%
4993   \hangindent##1\glstreeindent\relax
4994   \parindent##1\glstreeindent\relax
4995   \ifnum##1=1\relax
4996     \glssubentryitem{##2}%
4997   \fi
```

```

4998 \glstarget{##2}{\strut}%
4999 ##4\glspostdescription\space ##6\par}%
    Vertical gap between groups is the same as that used by indices:
5000 \renewcommand*\glsgroupskip{\indexspace}%
5001 }

```

treenonamegroup Like the `treenoname` style but the glossary groups have headings.

```

5002 \newglossarystyle{treenonamegroup}{%
    Base it on the glostyletreenoname style:
5003 \glossarystyle{treenoname}%
    Give each group a heading:
5004 \renewcommand{\glsgroupheading}[1]{\par
5005 \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
5006 }

```

treenonamehypergroup The `treenonamehypergroup` style is like the `treenonamegroup` style, but has a set of links to the groups at the start of the glossary.

```

5007 \newglossarystyle{treenonamehypergroup}{%
    Base it on the glostyletreenoname style:
5008 \glossarystyle{treenoname}%
    Put navigation links to the groups at the start of the theglossary environment:
5009 \renewcommand*\glossaryheader}{%
5010 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
    Each group has a heading (in bold with a target) followed by a vertical gap):
5011 \renewcommand*\glsgroupheading}[1]{%
5012 \par\noindent
5013 \textbf{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}\par
5014 \indexspace}%
5015 }

```

\glsssetwidest `\glsssetwidest[level]{text}` sets the widest text for the given level. It is used by the `alttree` glossary styles to determine the indentation of each level.

```

5016 \newcommand*\glsssetwidest}[2][0]{%
5017 \expandafter\def\csname @glswidestname\romannumeral#1\endcsname{%
5018 #2}%
5019 }

```

\@glswidestname Initialise `\@glswidestname`.

```

5020 \newcommand*\@glswidestname{}

```

alttree The `alttree` glossary style is similar in style to the `tree` style, but the indentation is obtained from the width of `\@glswidestname` which is set using `\glsssetwidest`.

```

5021 \newglossarystyle{alttree}{%

```

Redefine theglossary environment.

```
5022 \renewenvironment{theglossary}%  
5023   {\def\@gls@prevlevel{-1}%  
5024    \mbox{}\par}%  
5025   {\par}%
```

Set the header and group headers to nothing.

```
5026 \renewcommand*\glossaryheader{}%  
5027 \renewcommand*\glsgroupheading}[1]{}%
```

Redefine the way that the level 0 entries are displayed.

```
5028 \renewcommand{\glossaryentryfield}[5]{%
```

If the level hasn't changed, keep the same settings, otherwise change `\glstreeindent` accordingly.

```
5029   \ifnum\@gls@prevlevel=0\relax  
5030   \else
```

Find out how big the indentation should be by measuring the widest entry.

```
5031     \settowidth{\glstreeindent}{\textbf{\@glswidestname\space}}%
```

Set the hangindent and paragraph indent.

```
5032     \hangindent\glstreeindent  
5033     \parindent\glstreeindent  
5034   \fi
```

Put the name to the left of the paragraph block.

```
5035   \makebox[Opt][r]{\makebox[\glstreeindent][l]{%  
5036     \glsentryitem{##1}\textbf{\glstarget{##1}{##2}}}}%
```

If the symbol is missing, ignore it, otherwise put it in brackets.

```
5037   \ifx\relax##4\relax  
5038   \else  
5039     (##4)\space  
5040   \fi
```

Do the description followed by the description terminator and location list.

```
5041   ##3\glspostdescription \space ##5\par
```

Set the previous level to 0.

```
5042   \def\@gls@prevlevel{0}%  
5043 }%
```

Redefine the way sub-entries are displayed.

```
5044 \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.

```
5045   \ifnum##1=1\relax  
5046     \glssubentryitem{##2}%  
5047   \fi
```

If the level hasn't changed, keep the same settings, otherwise adjust `\glstreeindent` accordingly.

```

5048     \ifnum\@gls@prevlevel=##1\relax
5049     \else
        Compute the widest entry for this level, or for level 0 if not defined for this level.
        Store in \gls@tmplen
5050     \@ifundefined{@glswidestname\romannumeral##1}{%
5051     \settowidth{\gls@tmplen}{\textbf{\@glswidestname\space}}}{%
5052     \settowidth{\gls@tmplen}{\textbf{%
5053     \csname @glswidestname\romannumeral##1\endcsname\space}}}%
        Determine if going up or down a level
5054     \ifnum\@gls@prevlevel<##1\relax
        Depth has increased, so add the width of the widest entry to \glstreeindent.
5055     \setlength\glstreeindent\gls@tmplen
5056     \addtolength\glstreeindent\parindent
5057     \parindent\glstreeindent
5058     \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.
5059     \@ifundefined{@glswidestname\romannumeral\@gls@prevlevel}{%
5060     \settowidth{\glstreeindent}{\textbf{%
5061     \@glswidestname\space}}}{%
5062     \settowidth{\glstreeindent}{\textbf{%
5063     \csname @glswidestname\romannumeral\@gls@prevlevel
5064     \endcsname\space}}}%
        Subtract this length from the previous level's paragraph indent and set to
        \glstreeindent.
5065     \addtolength\parindent{-\glstreeindent}%
5066     \setlength\glstreeindent\parindent
5067     \fi
5068     \fi
        Set the hanging indentation.
5069     \hangindent\glstreeindent
        Put the name to the left of the paragraph block
5070     \makebox[Opt][r]{\makebox[\gls@tmplen][l]{%
5071     \textbf{\glstarget{##2}{##3}}}}%
        If the symbol is missing, ignore it, otherwise put it in brackets.
5072     \ifx##5\relax\relax
5073     \else
5074     (##5)\space
5075     \fi
        Do the description followed by the description terminator and location list.
5076     ##4\glspostdescription\space ##6\par

```

Set the previous level macro to the current level.

```
5077 \def\@gls@prevlevel{##1}%  
5078 }%
```

Vertical gap between groups is the same as that used by indices:

```
5079 \renewcommand*\glsgroupskip{\indexspace}%  
5080 }
```

almtreegroup Like the `almtree` style but the glossary groups have headings.

```
5081 \newglossarystyle{almtreegroup}{%
```

Base it on the `glostylealmtree` style:

```
5082 \glossarystyle{almtree}%
```

Give each group a heading.

```
5083 \renewcommand{\glsgroupheading}[1]{\par  
5084 \def\@gls@prevlevel{-1}%  
5085 \hangindent0pt\relax  
5086 \parindent0pt\relax  
5087 \textbf{\glsgetgrouptitle{##1}}\par\indexspace}%  
5088 }
```

almtreehypergroup The `almtreehypergroup` style is like the `almtreegroup` style, but has a set of links to the groups at the start of the glossary.

```
5089 \newglossarystyle{almtreehypergroup}{%
```

Base it on the `glostylealmtree` style:

```
5090 \glossarystyle{almtree}%
```

Put the navigation links in the header

```
5091 \renewcommand*\glossaryheader{%  
5092 \par  
5093 \def\@gls@prevlevel{-1}%  
5094 \hangindent0pt\relax  
5095 \parindent0pt\relax  
5096 \textbf{\glsnavigation}\par\indexspace}%
```

Put a `hypertarget` at the start of each group

```
5097 \renewcommand*\glsgroupheading}[1]{%  
5098 \par  
5099 \def\@gls@prevlevel{-1}%  
5100 \hangindent0pt\relax  
5101 \parindent0pt\relax  
5102 \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par  
5103 \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.

```

5104 \NeedsTeXFormat{LaTeX2e}
5105 \ProvidesPackage{glossaries-compatible-207}[2011/04/02 v1.0 (NLCT)]

```

`\GlsAddXdyAttribute` Adds an attribute in old format.

```

5106 \ifglxsindy
5107 \renewcommand*\GlsAddXdyAttribute[1]{%
5108 \edef\xdyattributes{\xdyattributes ^^J \string"#1\string"}%
5109 \expandafter\toks@\expandafter{\@xdylocref}%
5110 \edef\@xdylocref{\the\toks@ ^^J%
5111 (markup-locref
5112 :open \string"\string~n\string\setentrycounter
5113 {\noexpand\glscounter}%
5114 \expandafter\string\csname#1\endcsname
5115 \expandafter@gobble\string\{\string" ^^J
5116 :close \string"\expandafter@gobble\string\}\string" ^^J
5117 :attr \string"#1\string")}}

```

Only has an effect before `\writeist`:

```

5118 \fi

```

`\GlsAddXdyCounters`

```

5119 \renewcommand*\GlsAddXdyCounters[1]{%
5120 \GlossariesWarning{\string\GlsAddXdyCounters\space not available
5121 in compatibility mode.}%
5122 }

```

Add predefined attributes

```

5123 \GlsAddXdyAttribute{glnumberformat}
5124 \GlsAddXdyAttribute{textrm}
5125 \GlsAddXdyAttribute{textsf}
5126 \GlsAddXdyAttribute{texttt}
5127 \GlsAddXdyAttribute{textbf}
5128 \GlsAddXdyAttribute{textmd}
5129 \GlsAddXdyAttribute{textit}
5130 \GlsAddXdyAttribute{textup}
5131 \GlsAddXdyAttribute{textsl}
5132 \GlsAddXdyAttribute{textsc}
5133 \GlsAddXdyAttribute{emph}
5134 \GlsAddXdyAttribute{glshypernumber}
5135 \GlsAddXdyAttribute{hyper rm}
5136 \GlsAddXdyAttribute{hypersf}
5137 \GlsAddXdyAttribute{hypertt}
5138 \GlsAddXdyAttribute{hyperbf}
5139 \GlsAddXdyAttribute{hypermd}
5140 \GlsAddXdyAttribute{hyperit}
5141 \GlsAddXdyAttribute{hyperup}
5142 \GlsAddXdyAttribute{hypersl}
5143 \GlsAddXdyAttribute{hypersc}
5144 \GlsAddXdyAttribute{hyperemph}

```

`\GlsAddXdyLocation` Restore v2.07 definition:

```
5145 \ifglxindy
5146   \renewcommand*{\GlsAddXdyLocation}[2]{%
5147     \edef\xdyuserlocationdefs{%
5148       \xdyuserlocationdefs ^^J%
5149       (define-location-class \string"#1\string"^^J\space\space
5150       \space(#2))
5151     }%
5152     \edef\xdyuserlocationnames{%
5153       \xdyuserlocationnames^^J\space\space\space
5154       \string"#1\string"}%
5155   }
5156 \fi
```

`\do@wrglossary`

```
5157 \renewcommand{\do@wrglossary}[1]{%
  Determine whether to use xindy or makeindex syntax
5158 \ifglxindy
  Need to determine if the formatting information starts with a ( or ) indicating a
  range.
5159 \expandafter\@glo@check@mkidxrangechar\@glsnumberformat\@nil
5160 \def\@glo@range{}%
5161 \expandafter\if\@glo@prefix(\relax
5162   \def\@glo@range{:open-range}%
5163   \else
5164     \expandafter\if\@glo@prefix)\relax
5165     \def\@glo@range{:close-range}%
5166   \fi
5167   \fi

  Get the location and escape any special characters
5168 \protected@edef\@glslocref{\theglslentrycounter}%
5169 \@gls@checkmkidxchars\@glslocref

  Write to the glossary file using xindy syntax.
5170 \glossary[\csname glo@#1@type\endcsname]{%
5171 (indexentry :tkey (\csname glo@#1@index\endcsname)
5172   :locref \string"\@glslocref\string" %
5173   :attr \string"\@glo@suffix\string" \@glo@range
5174 )
5175 }%
5176 \else

  Convert the format information into the format required for makeindex
5177 \set@glo@numformat\@glo@numfmt\@gls@counter\@glsnumberformat

  Write to the glossary file using makeindex syntax.
5178 \glossary[\csname glo@#1@type\endcsname]{%
5179 \string\glossaryentry{\csname glo@#1@index\endcsname
5180   \@gls@encapchar\@glo@numfmt}{\theglslentrycounter}}%
```

```
5181 \fi
5182 }
```

`\@set@glo@numformat` Only had 3 arguments in v2.07

```
5183 \def\@set@glo@numformat#1#2#3{%
5184   \expandafter\@glo@check@mkidxrangear#3\@nil
5185   \protected@edef#1{%
5186     \@glo@prefix setentrycounter[]{#2}%
5187     \expandafter\string\csname\@glo@suffix\endcsname
5188   }%
5189   \@gls@checkmkidxchars#1%
5190 }
```

`\writeist` Redefine `\writeist` back to the way it was in v2.07, but change `\istfile` to `\glswrite`.

```
5191 \ifglxsindy
5192   \def\writeist{%
5193     \openout\glswrite=\istfilename
5194     \write\glswrite{;; xindy style file created by the glossaries
5195       package in compatible-2.07 mode}%
5196     \write\glswrite{;; for document '\jobname' on
5197       \the\year-\the\month-\the\day}%
5198     \write\glswrite{^^J; required styles^^J}
5199     \@for\@xdystyle:=\@xdyrequiredstyles\do{%
5200       \ifx\@xdystyle\@empty
5201         \else
5202           \protected@write\glswrite{{(require
5203             \string"\@xdystyle.xdy\string")}%
5204         \fi
5205       }%
5206     \write\glswrite{^^J%
5207       ; list of allowed attributes (number formats)^^J}%
5208     \write\glswrite{(define-attributes ((\@xdyattributes)))}%
5209     \write\glswrite{^^J; user defined alphabets^^J}%
5210     \write\glswrite{\@xdyuseralphabets}%
5211     \write\glswrite{^^J; location class definitions^^J}%
5212     \protected@edef\@gls@roman{\@roman{0}\string"
5213       \string"roman-numbers-lowercase\string" :sep \string"}%
5214     \@onelevel@sanitize\@gls@roman
5215     \edef\@tmp{\string" \string"roman-numbers-lowercase\string"
5216       :sep \string"}%
5217     \@onelevel@sanitize\@tmp
5218     \ifx\@tmp\@gls@roman
5219       \write\glswrite{(define-location-class
5220         \string"roman-page-numbers\string"^^J\space\space\space
5221         (\string"roman-numbers-lowercase\string")
5222         :min-range-length \@glsminrange)}%
5223     \else
5224       \write\glswrite{(define-location-class
5225         \string"roman-page-numbers\string"^^J\space\space\space
```

```

5226         (:sep "\@gls@roman")
5227         :min-range-length \@glsminrange)}%
5228 \fi
5229 \write\glswrite{(define-location-class
5230   \string"Roman-page-numbers\string"^^J\space\space\space
5231   (\string"roman-numbers-uppercase\string")
5232   :min-range-length \@glsminrange)}%
5233 \write\glswrite{(define-location-class
5234   \string"arabic-page-numbers\string"^^J\space\space\space
5235   (\string"arabic-numbers\string")
5236   :min-range-length \@glsminrange)}%
5237 \write\glswrite{(define-location-class
5238   \string"alpha-page-numbers\string"^^J\space\space\space
5239   (\string"alpha\string")
5240   :min-range-length \@glsminrange)}%
5241 \write\glswrite{(define-location-class
5242   \string"Alpha-page-numbers\string"^^J\space\space\space
5243   (\string"ALPHA\string")
5244   :min-range-length \@glsminrange)}%
5245 \write\glswrite{(define-location-class
5246   \string"Appendix-page-numbers\string"^^J\space\space\space
5247   (\string"ALPHA\string"
5248     :sep \string"\@glsAlpha compositor\string"
5249     \string"arabic-numbers\string")
5250   :min-range-length \@glsminrange)}%
5251 \write\glswrite{(define-location-class
5252   \string"arabic-section-numbers\string"^^J\space\space\space
5253   (\string"arabic-numbers\string"
5254     :sep \string"\glscompositor\string"
5255     \string"arabic-numbers\string")
5256   :min-range-length \@glsminrange)}%
5257 \write\glswrite{^^J; user defined location classes}%
5258 \write\glswrite{\@xdyuserlocationdefs}%
5259 \write\glswrite{^^J; define cross-reference class^^J}%
5260 \write\glswrite{(define-crossref-class \string"see\string"
5261   :unverified )}%
5262 \write\glswrite{(markup-crossref-list
5263   :class \string"see\string"^^J\space\space\space
5264   :open \string"\string\glsseeformat\string"
5265   :close \string"{}\string")}%
5266 \write\glswrite{^^J; define the order of the location classes}%
5267 \write\glswrite{(define-location-class-order
5268   (\@xdylocationclassorder))}%
5269 \write\glswrite{^^J; define the glossary markup^^J}%
5270 \write\glswrite{(markup-index^^J\space\space\space
5271   :open \string"\string
5272     \glossarysection[\string\glossarytoctitle]{\string
5273     \glossarytitle}\string\glossarypreamble\string~n\string\begin
5274     {theglossary}\string\glossaryheader\string~n\string" ^^J\space
5275     \space\space:close \string"\expandafter\@gobble

```

```

5276     \string%\string~n\string
5277     \end{theglossary}\string\glossarypostamble
5278     \string~n\string" ^^J\space\space\space
5279     :tree)}}%
5280 \write\glswrite{(markup-letter-group-list
5281   :sep \string"\string\glsgroupskip\string~n\string"}}%
5282 \write\glswrite{(markup-indexentry
5283   :open \string"\string\relax \string\glsresetentrylist
5284     \string~n\string"}}%
5285 \write\glswrite{(markup-locclass-list :open
5286   \string"\glsopenbrace\string\glossaryentrynumbers
5287     \glsopenbrace\string\relax\space \string"^^J\space\space\space
5288     :sep \string", \string"
5289     :close \string"\glsclosebrace\glsclosebrace\string"}}%
5290 \write\glswrite{(markup-locref-list
5291   :sep \string"\string\delimN\space\string"}}%
5292 \write\glswrite{(markup-range
5293   :sep \string"\string\delimR\space\string"}}%
5294 \@onelevel@sanitize\gls@suffixF
5295 \@onelevel@sanitize\gls@suffixFF
5296 \ifx\gls@suffixF\@empty
5297 \else
5298   \write\glswrite{(markup-range
5299     :close "\gls@suffixF" :length 1 :ignore-end)}}%
5300 \fi
5301 \ifx\gls@suffixFF\@empty
5302 \else
5303   \write\glswrite{(markup-range
5304     :close "\gls@suffixFF" :length 2 :ignore-end)}}%
5305 \fi
5306 \write\glswrite{^^J; define format to use for locations^^J}%
5307 \write\glswrite{\@xdylocref}%
5308 \write\glswrite{^^J; define letter group list format^^J}%
5309 \write\glswrite{(markup-letter-group-list
5310   :sep \string"\string\glsgroupskip\string~n\string"}}%
5311 \write\glswrite{^^J; letter group headings^^J}%
5312 \write\glswrite{(markup-letter-group
5313   :open-head \string"\string\glsgroupheading
5314     \glsopenbrace\string"^^J\space\space\space
5315     :close-head \string"\glsclosebrace\string"}}%
5316 \write\glswrite{^^J; additional letter groups^^J}%
5317 \write\glswrite{\@xdylettergroups}%
5318 \write\glswrite{^^J; additional sort rules^^J}
5319 \write\glswrite{\@xdysortrules}%
5320 \noist}
5321 \else
5322 \edef\@gls@actualchar{\string?}
5323 \edef\@gls@encapchar{\string|}
5324 \edef\@gls@levelchar{\string!}
5325 \edef\@gls@quotechar{\string"}

```

```

5326 \def\writeist{\relax
5327   \openout\glswrite=\istfilename
5328   \write\glswrite{\expandafter\@gobble\string\% makeindex style file
5329     created by the glossaries package}
5330   \write\glswrite{\expandafter\@gobble\string\% for document
5331     'jobname' on \the\year-\the\month-\the\day}
5332   \write\glswrite{actual '\@gls@actualchar'}
5333   \write\glswrite{encap '\@gls@encapchar'}
5334   \write\glswrite{level '\@gls@levelchar'}
5335   \write\glswrite{quote '\@gls@quotear'}
5336   \write\glswrite{keyword \string\string\glossaryentry\string}
5337   \write\glswrite{preamble \string\string\glossarysection[\string
5338     \glossarytoctitle]{\string\glossarytitle}\string
5339     \glossarypreamble\string\n\string\begin{theglossary}\string
5340     \glossaryheader\string\n\string}
5341   \write\glswrite{postamble \string\string%\string\n\string
5342     \end{theglossary}\string\glossarypostamble\string\n
5343     \string}
5344   \write\glswrite{group_skip \string\string\glsgroupskip\string\n
5345     \string}
5346   \write\glswrite{item_0 \string\string%\string\n\string}
5347   \write\glswrite{item_1 \string\string%\string\n\string}
5348   \write\glswrite{item_2 \string\string%\string\n\string}
5349   \write\glswrite{item_01 \string\string%\string\n\string}
5350   \write\glswrite{item_x1
5351     \string\string\relax \string\glsresetentrylist\string\n
5352     \string}
5353   \write\glswrite{item_12 \string\string%\string\n\string}
5354   \write\glswrite{item_x2
5355     \string\string\relax \string\glsresetentrylist\string\n
5356     \string}
5357   \write\glswrite{delim_0 \string\string\{\string
5358     \glossaryentrynumbers\string\{\string\relax \string}
5359   \write\glswrite{delim_1 \string\string\{\string
5360     \glossaryentrynumbers\string\{\string\relax \string}
5361   \write\glswrite{delim_2 \string\string\{\string
5362     \glossaryentrynumbers\string\{\string\relax \string}
5363   \write\glswrite{delim_t \string\string\}\string}\string}
5364   \write\glswrite{delim_n \string\string\delimN \string}
5365   \write\glswrite{delim_r \string\string\delimR \string}
5366   \write\glswrite{headings_flag 1}
5367   \write\glswrite{heading_prefix
5368     \string\string\glsgroupheading\string\{\string}
5369   \write\glswrite{heading_suffix
5370     \string\string}\string\relax
5371     \string\glsresetentrylist \string}
5372   \write\glswrite{symhead_positive \string\glsymbols\string}
5373   \write\glswrite{numhead_positive \string\glsnumbers\string}
5374   \write\glswrite{page_compositor \string\glscompositor\string}
5375   \@gls@escbsdq\gls@suffixF

```

```

5376 \gls@escbsdq\gls@suffixFF
5377 \ifx\gls@suffixF\@empty
5378 \else
5379 \write\glswrite{suffix_2p \string"\gls@suffixF\string"}
5380 \fi
5381 \ifx\gls@suffixFF\@empty
5382 \else
5383 \write\glswrite{suffix_3p \string"\gls@suffixFF\string"}
5384 \fi
5385 \noist
5386 }
5387 \fi

```

\noist

```
5388 \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.

```
5389 \NeedsTeXFormat{LaTeX2e}
```

Package version number now in line with main glossaries package number but will only be updated when `glossaries-accsupp.sty` is modified.

```
5390 \ProvidesPackage{glossaries-accsupp}[2011/04/02 v3.0 (NLCT)]
```

```
5391 Experimental glossaries accessibility]
```

Pass all options to `glossaries`:

```
5392 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{glossaries}}
```

Process options:

```
5393 \ProcessOptions
```

Required packages:

```
5394 \RequirePackage{glossaries}
```

```
5395 \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:

```
5396 \define@key{glossentry}{access}{%
```

```
5397 \def\@glo@access{#1}%
```

```
5398 }
```

textaccess The replacement text corresponding to the text key:
5399 `\define@key{glossentry}{textaccess}{%`
5400 `\def\@glo@textaccess{#1}%`
5401 `}`

firstaccess The replacement text corresponding to the first key:
5402 `\define@key{glossentry}{firstaccess}{%`
5403 `\def\@glo@firstaccess{#1}%`
5404 `}`

pluralaccess The replacement text corresponding to the plural key:
5405 `\define@key{glossentry}{pluralaccess}{%`
5406 `\def\@glo@pluralaccess{#1}%`
5407 `}`

firstpluralaccess The replacement text corresponding to the firstplural key:
5408 `\define@key{glossentry}{firstpluralaccess}{%`
5409 `\def\@glo@firstpluralaccess{#1}%`
5410 `}`

symbolaccess The replacement text corresponding to the symbol key:
5411 `\define@key{glossentry}{symbolaccess}{%`
5412 `\def\@glo@symbolaccess{#1}%`
5413 `}`

symbolpluralaccess The replacement text corresponding to the symbolplural key:
5414 `\define@key{glossentry}{symbolpluralaccess}{%`
5415 `\def\@glo@symbolpluralaccess{#1}%`
5416 `}`

descriptionaccess The replacement text corresponding to the description key:
5417 `\define@key{glossentry}{descriptionaccess}{%`
5418 `\def\@glo@descaccess{#1}%`
5419 `}`

descriptionpluralaccess The replacement text corresponding to the descriptionplural key:
5420 `\define@key{glossentry}{descriptionpluralaccess}{%`
5421 `\def\@glo@descpluralaccess{#1}%`
5422 `}`

shortaccess The replacement text corresponding to the short key:
5423 `\define@key{glossentry}{shortaccess}{%`
5424 `\def\@glo@shortaccess{#1}%`
5425 `}`

shortpluralaccess The replacement text corresponding to the shortplural key:
5426 `\define@key{glossentry}{shortpluralaccess}{%`
5427 `\def\@glo@shortpluralaccess{#1}%`
5428 `}`

`longaccess` The replacement text corresponding to the long key:

```
5429 \define@key{glossentry}{longaccess}{%
5430 \def\@glo@longaccess{#1}%
5431 }
```

`longpluralaccess` The replacement text corresponding to the longplural key:

```
5432 \define@key{glossentry}{longpluralaccess}{%
5433 \def\@glo@longpluralaccess{#1}%
5434 }
```

There are no equivalent keys for the `user1...user6` keys. The replacement text would have to be explicitly put in the value, e.g., `user1={\glsaccsupp{inches}{in}}`.

`\@gls@noaccess` Indicates that no replacement text has been provided.

```
5435 \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):

```
5436 \let\@gls@oldnewglossaryentryprehook\@newglossaryentryprehook
5437 \renewcommand*\@newglossaryentryprehook{%
5438 \@gls@oldnewglossaryentryprehook
5439 \def\@glo@access{\@glo@symbol}%
```

Initialise the other keys:

```
5440 \def\@glo@textaccess{\@glo@access}%
5441 \def\@glo@firstaccess{\@glo@access}%
5442 \def\@glo@pluralaccess{\@glo@textaccess}%
5443 \def\@glo@firstpluralaccess{\@glo@pluralaccess}%
5444 \def\@glo@symbolaccess{\relax}%
5445 \def\@glo@symbolpluralaccess{\@glo@symbolaccess}%
5446 \def\@glo@descaccess{\relax}%
5447 \def\@glo@descpluralaccess{\@glo@descaccess}%
5448 \def\@glo@shortaccess{\relax}%
5449 \def\@glo@shortpluralaccess{\@glo@shortaccess}%
5450 \def\@glo@longaccess{\relax}%
5451 \def\@glo@longpluralaccess{\@glo@longaccess}%
5452 }
```

Add to the end hook:

```
5453 \let\@gls@oldnewglossaryentryposthook\@newglossaryentryposthook
5454 \renewcommand*\@newglossaryentryposthook{%
5455 \@gls@oldnewglossaryentryposthook
```

Store the access information:

```
5456 \expandafter
5457 \protected@xdef\csname glo@\@glo@label @access\endcsname{%
5458 \@glo@access}%
5459 \expandafter
5460 \protected@xdef\csname glo@\@glo@label @textaccess\endcsname{%
5461 \@glo@textaccess}%
```

```

5462 \expandafter
5463   \protected@xdef\csname glo@\@glo@label @firstaccess\endcsname{%
5464     \@glo@firstaccess}%
5465 \expandafter
5466   \protected@xdef\csname glo@\@glo@label @pluralaccess\endcsname{%
5467     \@glo@pluralaccess}%
5468 \expandafter
5469   \protected@xdef\csname glo@\@glo@label @firstpluralaccess\endcsname{%
5470     \@glo@firstpluralaccess}%
5471 \expandafter
5472   \protected@xdef\csname glo@\@glo@label @symbolaccess\endcsname{%
5473     \@glo@symbolaccess}%
5474 \expandafter
5475   \protected@xdef\csname glo@\@glo@label @symbolpluralaccess\endcsname{%
5476     \@glo@symbolpluralaccess}%
5477 \expandafter
5478   \protected@xdef\csname glo@\@glo@label @descaccess\endcsname{%
5479     \@glo@descaccess}%
5480 \expandafter
5481   \protected@xdef\csname glo@\@glo@label @descpluralaccess\endcsname{%
5482     \@glo@descpluralaccess}%
5483 \expandafter
5484   \protected@xdef\csname glo@\@glo@label @shortaccess\endcsname{%
5485     \@glo@shortaccess}%
5486 \expandafter
5487   \protected@xdef\csname glo@\@glo@label @shortpluralaccess\endcsname{%
5488     \@glo@shortpluralaccess}%
5489 \expandafter
5490   \protected@xdef\csname glo@\@glo@label @longaccess\endcsname{%
5491     \@glo@longaccess}%
5492 \expandafter
5493   \protected@xdef\csname glo@\@glo@label @longpluralaccess\endcsname{%
5494     \@glo@longpluralaccess}%
5495 }

```

5.2 Accessing Replacement Text

`\glsentryaccess` Get the value of the access key for the entry with the given label:

```

5496 \newcommand*{\glsentryaccess}[1]{%
5497   \csname glo@#1@access\endcsname
5498 }

```

`\glsentrytextaccess` Get the value of the textaccess key for the entry with the given label:

```

5499 \newcommand*{\glsentrytextaccess}[1]{%
5500   \csname glo@#1@textaccess\endcsname
5501 }

```

`\glsentryfirstaccess` Get the value of the firstaccess key for the entry with the given label:

```

5502 \newcommand*{\glsentryfirstaccess}[1]{%

```

```
5503 \csname glo@#1@firstaccess\endcsname
5504 }
```

`\glsentrypluralaccess` Get the value of the pluralaccess key for the entry with the given label:

```
5505 \newcommand*{\glsentrypluralaccess}[1]{%
5506 \csname glo@#1@pluralaccess\endcsname
5507 }
```

`\glsentryfirstpluralaccess` Get the value of the firstpluralaccess key for the entry with the given label:

```
5508 \newcommand*{\glsentryfirstpluralaccess}[1]{%
5509 \csname glo@#1@firstpluralaccess\endcsname
5510 }
```

`\glsentrysymbolaccess` Get the value of the symbolaccess key for the entry with the given label:

```
5511 \newcommand*{\glsentrysymbolaccess}[1]{%
5512 \csname glo@#1@symbolaccess\endcsname
5513 }
```

`\glsentrysymbolpluralaccess` Get the value of the symbolpluralaccess key for the entry with the given label:

```
5514 \newcommand*{\glsentrysymbolpluralaccess}[1]{%
5515 \csname glo@#1@symbolpluralaccess\endcsname
5516 }
```

`\glsentrydescaccess` Get the value of the descriptionaccess key for the entry with the given label:

```
5517 \newcommand*{\glsentrydescaccess}[1]{%
5518 \csname glo@#1@descaccess\endcsname
5519 }
```

`\glsentrydescpluralaccess` Get the value of the descriptionpluralaccess key for the entry with the given label:

```
5520 \newcommand*{\glsentrydescpluralaccess}[1]{%
5521 \csname glo@#1@descaccess\endcsname
5522 }
```

`\glsentryshortaccess` Get the value of the shortaccess key for the entry with the given label:

```
5523 \newcommand*{\glsentryshortaccess}[1]{%
5524 \csname glo@#1@shortaccess\endcsname
5525 }
```

`\glsentryshortpluralaccess` Get the value of the shortpluralaccess key for the entry with the given label:

```
5526 \newcommand*{\glsentryshortpluralaccess}[1]{%
5527 \csname glo@#1@shortpluralaccess\endcsname
5528 }
```

`\glsentrylongaccess` Get the value of the longaccess key for the entry with the given label:

```
5529 \newcommand*{\glsentrylongaccess}[1]{%
5530 \csname glo@#1@longaccess\endcsname
5531 }
```

```

\glsentrylongpluralaccess  Get the value of the longpluralaccess key for the entry with the given label:
5532 \newcommand*{\glsentrylongpluralaccess}[1]{%
5533   \csname glo@#1@longpluralaccess\endcsname
5534 }

\glsaccsupp  \glsaccsupp{<replacement text>}{<text>}

This can be redefined to use E or Alt instead of ActualText. (I don't have the
software to test the E or Alt options.)
5535 \newcommand*{\glsaccsupp}[2]{%
5536   \BeginAccSupp{ActualText=#1}#2\EndAccSupp{}}%
5537 }

\xglsaccsupp  Fully expands replacement text before calling \glsaccsupp
5538 \newcommand*{\xglsaccsupp}[2]{%
5539   \protected@edef\@gls@replacementtext{#1}%
5540   \expandafter\glsaccsupp\expandafter{\@gls@replacementtext}{#2}%
5541 }

\glsnameaccessdisplay  Displays the first argument with the accessibility text for the entry with the label
                        given by the second argument (if set).
5542 \DeclareRobustCommand*{\glsnameaccessdisplay}[2]{%
5543   \protected@edef\@glo@access{\glsentryaccess{#2}}%
5544   \ifx\@glo@access\@gls@noaccess
5545     #1%
5546   \else
5547     \xglsaccsupp{\@glo@access}{#1}%
5548   \fi
5549 }

\glsstextaccessdisplay  As above but for the textaccess replacement text.
5550 \DeclareRobustCommand*{\glsstextaccessdisplay}[2]{%
5551   \protected@edef\@glo@access{\glsentrytextaccess{#2}}%
5552   \ifx\@glo@access\@gls@noaccess
5553     #1%
5554   \else
5555     \xglsaccsupp{\@glo@access}{#1}%
5556   \fi
5557 }

\glspluralaccessdisplay  As above but for the pluralaccess replacement text.
5558 \DeclareRobustCommand*{\glspluralaccessdisplay}[2]{%
5559   \protected@edef\@glo@access{\glsentrypluralaccess{#2}}%
5560   \ifx\@glo@access\@gls@noaccess
5561     #1%
5562   \else
5563     \xglsaccsupp{\@glo@access}{#1}%
5564   \fi
5565 }

```

`\glsfirstaccessdisplay` As above but for the `firstaccess` replacement text.

```
5566 \DeclareRobustCommand*\glsfirstaccessdisplay}[2]{%
5567   \protected@edef\@glo@access{\glsentryfirstaccess{#2}}%
5568   \ifx\@glo@access\@gls@noaccess
5569     #1%
5570   \else
5571     \xglsaccsupp{\@glo@access}{#1}%
5572   \fi
5573 }
```

`\glsfirstpluralaccessdisplay` As above but for the `firstpluralaccess` replacement text.

```
5574 \DeclareRobustCommand*\glsfirstpluralaccessdisplay}[2]{%
5575   \protected@edef\@glo@access{\glsentryfirstpluralaccess{#2}}%
5576   \ifx\@glo@access\@gls@noaccess
5577     #1%
5578   \else
5579     \xglsaccsupp{\@glo@access}{#1}%
5580   \fi
5581 }
```

`\glsymbolaccessdisplay` As above but for the `symbolaccess` replacement text.

```
5582 \DeclareRobustCommand*\glsymbolaccessdisplay}[2]{%
5583   \protected@edef\@glo@access{\glsentrysymbolaccess{#2}}%
5584   \ifx\@glo@access\@gls@noaccess
5585     #1%
5586   \else
5587     \xglsaccsupp{\@glo@access}{#1}%
5588   \fi
5589 }
```

`\glsymbolpluralaccessdisplay` As above but for the `symbolpluralaccess` replacement text.

```
5590 \DeclareRobustCommand*\glsymbolpluralaccessdisplay}[2]{%
5591   \protected@edef\@glo@access{\glsentrysymbolpluralaccess{#2}}%
5592   \ifx\@glo@access\@gls@noaccess
5593     #1%
5594   \else
5595     \xglsaccsupp{\@glo@access}{#1}%
5596   \fi
5597 }
```

`\glsdescriptionaccessdisplay` As above but for the `descriptionaccess` replacement text.

```
5598 \DeclareRobustCommand*\glsdescriptionaccessdisplay}[2]{%
5599   \protected@edef\@glo@access{\glsentrydescaccess{#2}}%
5600   \ifx\@glo@access\@gls@noaccess
5601     #1%
5602   \else
5603     \xglsaccsupp{\@glo@access}{#1}%
5604   \fi
5605 }
```

`\glsdescriptionpluralaccessdisplay` As above but for the `descriptionpluralaccess` replacement text.

```
5606 \DeclareRobustCommand*\glsdescriptionpluralaccessdisplay}[2]{%
5607   \protected@edef\@glo@access{\glsentrydescpluralaccess{#2}}%
5608   \ifx\@glo@access\@gls@noaccess
5609     #1%
5610   \else
5611     \xglsaccsupp{\@glo@access}{#1}%
5612   \fi
5613 }
```

`\glsshortaccessdisplay` As above but for the `shortaccess` replacement text.

```
5614 \DeclareRobustCommand*\glsshortaccessdisplay}[2]{%
5615   \protected@edef\@glo@access{\glsentryshortaccess{#2}}%
5616   \ifx\@glo@access\@gls@noaccess
5617     #1%
5618   \else
5619     \xglsaccsupp{\@glo@access}{#1}%
5620   \fi
5621 }
```

`\glsshortpluralaccessdisplay` As above but for the `shortpluralaccess` replacement text.

```
5622 \DeclareRobustCommand*\glsshortpluralaccessdisplay}[2]{%
5623   \protected@edef\@glo@access{\glsentryshortpluralaccess{#2}}%
5624   \ifx\@glo@access\@gls@noaccess
5625     #1%
5626   \else
5627     \xglsaccsupp{\@glo@access}{#1}%
5628   \fi
5629 }
```

`\glslongaccessdisplay` As above but for the `longaccess` replacement text.

```
5630 \DeclareRobustCommand*\glslongaccessdisplay}[2]{%
5631   \protected@edef\@glo@access{\glsentrylongaccess{#2}}%
5632   \ifx\@glo@access\@gls@noaccess
5633     #1%
5634   \else
5635     \xglsaccsupp{\@glo@access}{#1}%
5636   \fi
5637 }
```

`\glslongpluralaccessdisplay` As above but for the `longpluralaccess` replacement text.

```
5638 \DeclareRobustCommand*\glslongpluralaccessdisplay}[2]{%
5639   \protected@edef\@glo@access{\glsentrylongpluralaccess{#2}}%
5640   \ifx\@glo@access\@gls@noaccess
5641     #1%
5642   \else
5643     \xglsaccsupp{\@glo@access}{#1}%
5644   \fi
5645 }
```

`\glsaccessdisplay` Gets the replacement text corresponding to the named key given by the first argument and calls the appropriate command defined above.

```

5646 \DeclareRobustCommand*\glsaccessdisplay}[3]{%
5647 \@ifundefined{gls#1accessdisplay}%
5648 {%
5649   \PackageError{glossaries-accsupp}{No accessibility support
5650     for key ‘#1’}{%
5651   }%
5652   {%
5653     \csname gls#1accessdisplay\endcsname{#2}{#3}%
5654   }%
5655 }

```

`\@gls@` Redefine `\@gls@` to change the way the link text is defined

```

5656 \def\@gls@#1#2[#3]{%
5657 \glsdoifexists{#2}%
5658 {%
5659   \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

5660   \def\@gls@link@opts{#1}%
5661   \def\@gls@link@label{#2}%

```

Determine what the link text should be (this is stored in `\@glo@text`). This is no longer expanded.

```

5662   \ifglsused{#2}%
5663   {%
5664     \def\@glo@text{\csname gls@\@glo@type @display\endcsname
5665       {\gls@textaccessdisplay{\glsentrytext{#2}}{#2}}%
5666       {\gls@descriptionaccessdisplay{\glsentrydesc{#2}}{#2}}%
5667       {\gls@symbolaccessdisplay{\glsentrysymbol{#2}}{#2}}%
5668       {#3}}%
5669   }%
5670   {%
5671     \def\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
5672       {\gls@firstaccessdisplay{\glsentryfirst{#2}}{#2}}%
5673       {\gls@descriptionaccessdisplay{\glsentrydesc{#2}}{#2}}%
5674       {\gls@symbolaccessdisplay{\glsentrysymbol{#2}}{#2}}%
5675       {#3}}%
5676   }%

```

Call `\@gls@link`. If footnote package option has been used, suppress hyperlink for first use.

```

5677   \ifglsused{#2}%
5678   {%
5679     \@gls@link[#1]{#2}{\@glo@text}%
5680   }%
5681   {%
5682     \gls@checkisacronymlist\@glo@type
5683     \ifthenelse{\boolean{glsisacronymlist}\AND

```

```

5684     \boolean{glsacrfootnote}\) \OR\NOT\boolean{gls hyperfirst}}%
5685     {%
5686     \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
5687     }%
5688     {%
5689     \@gls@link[#1]{#2}{\@glo@text}%
5690     }%
5691     }%

```

Indicate that this entry has now been used

```

5692     \glsunset{#2}%
5693     }%
5694 }

```

\@Gls@

```

5695 \def\@Gls@#1#2[#3]{%
5696   \glsdoifexists{#2}%
5697   {%
5698     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

5699     \def\@gls@link@opts{#1}%
5700     \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

```

5701     \ifglsused{#2}%
5702     {%
5703     \def\@glo@text{\csname gls@\@glo@type @display\endcsname
5704     {\gls@textaccessdisplay{\Glsentrytext{#2}}{#2}}%
5705     {\glsdescriptionaccessdisplay{\glsentrydesc{#2}}{#2}}%
5706     {\glsymbolaccessdisplay{\glsentrysymbol{#2}}{#2}}%
5707     {#3}}%
5708     }%
5709     {%
5710     \def\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
5711     {\glsfirstaccessdisplay{\Glsentryfirst{#2}}{#2}}%
5712     {\glsdescriptionaccessdisplay{\glsentrydesc{#2}}{#2}}%
5713     {\glsymbolaccessdisplay{\glsentrysymbol{#2}}{#2}}%
5714     {#3}}%
5715     }%

```

Call \@gls@link. If footnote package option has been used, suppress hyperlink for first use.

```

5716     \ifglsused{#2}%
5717     {%
5718     \@gls@link[#1]{#2}{\@glo@text}%
5719     }%
5720     {%
5721     \gls@checkisacronymlist\@glo@type

```

```

5722 \ifthenelse{\(\boolean{@glsisacronymlist}\AND
5723 \boolean{glsacrfootnote}\) \OR\NOT\boolean{glshyperfirst}}%
5724 {%
5725 \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
5726 }%
5727 {%
5728 \@gls@link[#1]{#2}{\@glo@text}%
5729 }%
5730 }%

```

Indicate that this entry has now been used

```

5731 \glsunset{#2}%
5732 }%
5733 }

```

\@GLS@

```

5734 \def\@GLS@#1#2[#3]{%
5735 \glsdoifexists{#2}{%
5736 \edef\@glo@type{\glsentrytype{#2}}%
5737 \def\@gls@link@opts{#1}%
5738 \def\@gls@link@label{#2}%
5739 \ifglsused{#2}%
5740 {%
5741 \def\@glo@text{\csname gls@\@glo@type @display\endcsname
5742 {\gls@textaccessdisplay{\glsentrytext{#2}}{#2}}%
5743 {\gls@descriptionaccessdisplay{\glsentrydesc{#2}}{#2}}%
5744 {\gls@symbolaccessdisplay{\glsentrysymbol{#2}}{#2}}%
5745 {#3}}%
5746 }%
5747 {%
5748 \edef\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
5749 {\gls@firstaccessdisplay{\glsentryfirst{#2}}{#2}}%
5750 {\gls@descriptionaccessdisplay{\glsentrydesc{#2}}{#2}}%
5751 {\gls@symbolaccessdisplay{\glsentrysymbol{#2}}{#2}}%
5752 {#3}}%
5753 }%

```

Call \@gls@link If footnote package option has been used, suppress hyperlink for first use.

```

5754 \ifglsused{#2}%
5755 {%
5756 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
5757 }%
5758 {%
5759 \gls@checkisacronymlist\@glo@type
5760 \ifthenelse{\(\boolean{@glsisacronymlist}\AND
5761 \boolean{glsacrfootnote}\) \OR\NOT\boolean{glshyperfirst}}{%

```

```

5762     \@gls@link[#1,hyper=false]{#2}{\MakeUppercase{\@glo@text}}%
5763     }%
5764     {%
5765     \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
5766     }%
5767     }%

```

Indicate that this entry has now been used

```

5768     \glsunset{#2}%
5769     }%
5770 }

```

\@gls@pl@

```

5771 \def\@glspl@#1#2[#3]{%
5772 \glsdoifexists{#2}%
5773   {%
5774     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

5775     \def\@gls@link@opts{#1}%
5776     \def\@gls@link@label{#2}%

```

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

```

5777     \ifglsused{#2}%
5778     {%
5779     \def\@glo@text{\csname gls@\@glo@type @display\endcsname
5780       {\glspluralaccessdisplay{\glsentryplural{#2}}{#2}}%
5781       {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
5782       {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
5783       {#3}}%
5784     }%
5785     {%
5786     \def\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
5787       {\glsfirstpluralaccessdisplay{\glsentryfirstplural{#2}}{#2}}%
5788       {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
5789       {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
5790       {#3}}%
5791     }%

```

Call \@gls@link If footnote package option has been used, suppress hyperlink for first use.

```

5792     \ifglsused{#2}%
5793     {%
5794     \@gls@link[#1]{#2}{\@glo@text}%
5795     }%
5796     {%
5797     \gls@checkisacronymlist\@glo@type
5798     \ifthenelse{\(\boolean{\@glsisacronymlist}\AND
5799       \boolean{\glsacrfootnote}\) \OR\not\boolean{\glshyperfirst}}%
5800     {%
5801     \@gls@link[#1,hyper=false]{#2}{\@glo@text}%

```

```

5802     }%
5803     {%
5804     \@gls@link[#1]{#2}{\@glo@text}%
5805     }%
5806     }%

```

Indicate that this entry has now been used

```

5807     \glsunset{#2}%
5808     }%
5809 }

```

`\@Glspl@`

```

5810 \def\@Glspl@#1#2[#3]{%
5811   \glsdoifexists{#2}%
5812   {%
5813     \edef\@glo@type{\glsentrytype{#2}}%
5814     Save options in \@gls@link@opts and label in \@gls@link@label
5815     \def\@gls@link@opts{#1}%
5816     \def\@gls@link@label{#2}%
5817     Determine what the link text should be (this is stored in \@glo@text).
5818     \ifglsused{#2}%
5819     {%
5820       \def\@glo@text{\csname gls@\@glo@type @display\endcsname
5821         {\glspluralaccessdisplay{\Glsentryplural{#2}}{#2}}%
5822         {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
5823         {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
5824         {#3}}%
5825     }%
5826     {%
5827       \def\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
5828         {\glsfirstpluralaccessdisplay{\Glsentryfirstplural{#2}}{#2}}%
5829         {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
5830         {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
5831         {#3}}%
5832     }%
5833   }%
5834 }

```

Call `\@gls@link` If footnote package option has been used, suppress hyperlink for first use.

```

5831     \ifglsused{#2}%
5832     {%
5833     \@gls@link[#1]{#2}{\@glo@text}%
5834     }%
5835     {%
5836     \ifthenelse{\equal{\@glo@type}{\acronymtype}\and
5837       \boolean{glsacrfootnote}}%
5838     {%
5839     \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
5840     }%
5841     }%

```

```

5842     \@gls@link[#1]{#2}{\@glo@text}%
5843     }%
5844     }%

```

Indicate that this entry has now been used

```

5845     \glsunset{#2}%
5846     }%
5847 }

```

\@GLSp1@

```

5848 \def\@GLSp1@#1#2[#3]{%
5849   \glsdoifexists{#2}%
5850   {%
5851     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

5852     \def\@gls@link@opts{#1}%
5853     \def\@gls@link@label{#2}%

```

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

```

5854     \ifglsused{#2}%
5855     {%
5856       \def\@glo@text{\csname gls@\@glo@type @display\endcsname
5857         {\glspluralaccessdisplay{\glsentryplural{#2}}{#2}}%
5858         {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
5859         {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
5860         {#3}}%
5861     }%
5862     {%
5863       \def\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
5864         {\glsfirstpluralaccessdisplay{\glsentryfirstplural{#2}}{#2}}%
5865         {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
5866         {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
5867         {#3}}%
5868     }%

```

Call \@gls@link If footnote package option has been used, suppress hyperlink for first use.

```

5869     \ifglsused{#2}%
5870     {%
5871       \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
5872     }%
5873     {%
5874       \gls@checkisacronymlist\@glo@type
5875       \ifthenelse{\(\boolean{glsisacronymlist}\)AND
5876         \boolean{glsacrfootnote}\)\OR\not\boolean{glsHYPERFIRST}}%
5877       {%
5878         \@gls@link[#1,hyper=false]{#2}{\MakeUppercase{\@glo@text}}%
5879       }%
5880       {%
5881         \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%

```

```

5882     }%
5883     }%
    Indicate that this entry has now been used
5884     \glsunset{#2}%
5885     }%
5886 }

```

\@acrshort

```

5887 \def\@acrshort#1#2[#3]{%
5888   \glsdoifexists{#2}%
5889   {%
5890     \edef\@glo@type{\glsentrytype{#2}}%
    Determine what the link text should be (this is stored in \@glo@text)
5891     \def\@glo@text{%
5892       \glsshortaccessdisplay{\glsentryshort{#2}}{#2}%
5893     }%
    Call \@gls@link
5894     \@gls@link[#1]{#2}{\acronymfont{\@glo@text}#3}%
5895     }%
5896 }

```

\@Acrshort

```

5897 \def\@Acrshort#1#2[#3]{%
5898   \glsdoifexists{#2}%
5899   {%
5900     \edef\@glo@type{\glsentrytype{#2}}%
    Determine what the link text should be (this is stored in \@glo@text)
5901     \def\@glo@text{%
5902       \glsshortaccessdisplay{\Glsentryshort{#2}}{#2}%
5903     }%
    Call \@gls@link
5904     \@gls@link[#1]{#2}{\acronymfont{\@glo@text}#3}%
5905     }%
5906 }

```

\@ACRshort

```

5907 \def\@ACRshort#1#2[#3]{%
5908   \glsdoifexists{#2}%
5909   {%
5910     \edef\@glo@type{\glsentrytype{#2}}%
    Determine what the link text should be (this is stored in \@glo@text)
5911     \def\@glo@text{%
5912       \glsshortaccessdisplay{\MakeUppercase{\glsentryshort{#2}}}{#2}%
5913     }%

```

```

Call \@gls@link
5914   \@gls@link[#1]{#2}{\acronymfont{\@glo@text#3}}%
5915   }%
5916 }

\@acrlong
5917 \def\@acrlong#1#2[#3]{%
5918   \glsdoifexists{#2}%
5919   {%
5920     \edef\@glo@type{\glsentrytype{#2}}%
Determine what the link text should be (this is stored in \@glo@text)
5921     \def\@glo@text{%
5922       \glslongaccessdisplay{\glsentrylong{#2}}{#2}%
5923     }%
Call \@gls@link
5924     \@gls@link[#1]{#2}{\@glo@text#3}%
5925   }%
5926 }

\@Acrlong
5927 \def\@Acrlong#1#2[#3]{%
5928   \glsdoifexists{#2}%
5929   {%
5930     \edef\@glo@type{\glsentrytype{#2}}%
Determine what the link text should be (this is stored in \@glo@text)
5931     \def\@glo@text{%
5932       \glslongaccessdisplay{\Glsentrylong{#2}}{#2}%
5933     }%
Call \@gls@link
5934     \@gls@link[#1]{#2}{\@glo@text#3}%
5935   }%
5936 }

\@ACRlong
5937 \def\@ACRlong#1#2[#3]{%
5938   \glsdoifexists{#2}%
5939   {%
5940     \edef\@glo@type{\glsentrytype{#2}}%
Determine what the link text should be (this is stored in \@glo@text)
5941     \def\@glo@text{%
5942       \glslongaccessdisplay{\MakeUppercase{\glsentrylong{#2}}}{#2}%
5943     }%
Call \@gls@link
5944     \@gls@link[#1]{#2}{\@glo@text#3}%
5945   }%
5946 }

```

5.3 Displaying the Glossary

Entries within the glossary or list of acronyms are now formatted via `\accsuppglossaryentryfield` and `\accsuppglossarysubentryfield`.

`\@glossaryentryfield`

```
5947 \ifglxindy
5948 \renewcommand*{\@glossaryentryfield}{%
5949   \string\accsuppglossaryentryfield}
5950 \else
5951 \renewcommand*{\@glossaryentryfield}{%
5952   \string\accsuppglossaryentryfield}
5953 \fi
```

`\@glossarysubentryfield`

```
5954 \ifglxindy
5955 \renewcommand*{\@glossarysubentryfield}{%
5956   \string\accsuppglossarysubentryfield}
5957 \else
5958 \renewcommand*{\@glossarysubentryfield}{%
5959   \string\accsuppglossarysubentryfield}
5960 \fi
```

`\accsuppglossaryentryfield`

```
5961 \newcommand*{\accsuppglossaryentryfield}[5]{%
5962   \glossaryentryfield{#1}%
5963   {\glsnameaccessdisplay{#2}{#1}}%
5964   {\glsdescriptionaccessdisplay{#3}{#1}}%
5965   {\glsymbolaccessdisplay{#4}{#1}}{#5}%
5966 }
```

`\accsuppglossarysubentryfield`

```
5967 \newcommand*{\accsuppglossarysubentryfield}[6]{%
5968   \glossaryentryfield{#1}{#2}%
5969   {\glsnameaccessdisplay{#3}{#2}}%
5970   {\glsdescriptionaccessdisplay{#4}{#2}}%
5971   {\glsymbolaccessdisplay{#5}{#2}}{#6}%
5972 }
```

5.4 Acronyms

Use `\newacronymhook` to modify the key list to set the access text to the long version by default.

```
5973 \renewcommand*{\newacronymhook}{%
5974   \edef\@gls@keylist{shortaccess=\the\glslongtok,%
5975     \the\glskeylisttok}%
5976   \expandafter\glskeylisttok\expandafter{\@gls@keylist}%
5977 }
```

`\DefaultNewAcronymDef` Modify default style to use access text:

```
5978 \renewcommand*{\DefaultNewAcronymDef}{%
5979   \edef\@do@newglossaryentry{%
5980     \noexpand\newglossaryentry{\the\glslabeltok}%
5981     {%
5982       type=\acronymtype,%
5983       name={\the\glsshorttok},%
5984       description={\the\glslongtok},%
5985       descriptionaccess=\relax,
5986       text={\the\glsshorttok},%
5987       access={\noexpand\@glo@textaccess},%
5988       sort={\the\glsshorttok},%
5989       short={\the\glsshorttok},%
5990       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
5991       shortaccess={\the\glslongtok},%
5992       long={\the\glslongtok},%
5993       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5994       descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
5995       first={\noexpand\glslongaccessdisplay
5996         {\the\glslongtok}{\the\glslabeltok}\space
5997         (\noexpand\glsshortaccessdisplay
5998           {\the\glsshorttok}{\the\glslabeltok})},%
5999       plural={\the\glsshorttok\acrpluralsuffix},%
6000       firstplural={\noexpand\glslongpluralaccessdisplay
6001         {\noexpand\@glo@longpl}{\the\glslabeltok}\space
6002         (\noexpand\glsshortpluralaccessdisplay
6003           {\noexpand\@glo@shortpl}{\the\glslabeltok})},%
6004       firstaccess=\relax,
6005       firstpluralaccess=\relax,
6006       textaccess={\noexpand\@glo@shortaccess},%
6007       \the\glskeylisttok
6008     }%
6009   }%
6010   \@do@newglossaryentry
6011 }
```

`DescriptionFootnoteNewAcronymDef`

```
6012 \renewcommand*{\DescriptionFootnoteNewAcronymDef}{%
6013   \edef\@do@newglossaryentry{%
6014     \noexpand\newglossaryentry{\the\glslabeltok}%
6015     {%
6016       type=\acronymtype,%
6017       name={\noexpand\acronymfont{\the\glsshorttok}},%
6018       sort={\the\glsshorttok},%
6019       text={\the\glsshorttok},%
6020       short={\the\glsshorttok},%
6021       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6022       shortaccess={\the\glslongtok},%
6023       long={\the\glslongtok},%
6024       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
```

```

6025     access={\noexpand@glo@textaccess},%
6026     plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6027     symbol={\the\glslongtok},%
6028     symbolplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6029     firstpluralaccess=\relax,
6030     textaccess={\noexpand@glo@shortaccess},%
6031     \the\glskeylisttok
6032   }%
6033 }%
6034 \do@newglossaryentry
6035 }

```

\DescriptionNewAcronymDef

```

6036 \renewcommand*{\DescriptionNewAcronymDef}{%
6037   \edef\do@newglossaryentry{%
6038     \noexpand\newglossaryentry{\the\glslabeltok}%
6039     {%
6040       type=\acronymtype,%
6041       name={\noexpand
6042         \acronymfont{\the\glsshorttok}{\the\glslongtok}},%
6043       access={\noexpand@glo@textaccess},%
6044       sort={\the\glsshorttok},%
6045       short={\the\glsshorttok},%
6046       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6047       shortaccess={\the\glslongtok},%
6048       long={\the\glslongtok},%
6049       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6050       first={\the\glslongtok},%
6051       firstaccess=\relax,
6052       firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6053       text={\the\glsshorttok},%
6054       textaccess={\the\glslongtok},%
6055       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6056       symbol={\noexpand@glo@text},%
6057       symbolaccess={\noexpand@glo@textaccess},%
6058       symbolplural={\noexpand@glo@plural},%
6059       firstpluralaccess=\relax,
6060       textaccess={\noexpand@glo@shortaccess},%
6061       \the\glskeylisttok}%
6062   }%
6063   \do@newglossaryentry
6064 }

```

\FootnoteNewAcronymDef

```

6065 \renewcommand*{\FootnoteNewAcronymDef}{%
6066   \edef\do@newglossaryentry{%
6067     \noexpand\newglossaryentry{\the\glslabeltok}%
6068     {%
6069       type=\acronymtype,%
6070       name={\noexpand\acronymfont{\the\glsshorttok}},%

```

```

6071     sort={\the\glsshorttok},%
6072     text={\the\glsshorttok},%
6073     textaccess={\the\glslongtok},%
6074     access={\noexpand\@glo@textaccess},%
6075     plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6076     short={\the\glsshorttok},%
6077     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6078     long={\the\glslongtok},%
6079     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6080     description={\the\glslongtok},%
6081     descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6082     \the\glskeylisttok
6083   }%
6084 }%
6085 \do@newglossaryentry
6086 }

```

\SmallNewAcronymDef

```

6087 \renewcommand*{\SmallNewAcronymDef}{%
6088   \edef\do@newglossaryentry{%
6089     \noexpand\newglossaryentry{\the\glslabeltok}%
6090     {%
6091       type=\acronymtype,%
6092       name={\noexpand\acronymfont{\the\glsshorttok}},%
6093       access={\noexpand\@glo@symbolaccess},%
6094       sort={\the\glsshorttok},%
6095       short={\the\glsshorttok},%
6096       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6097       shortaccess={\the\glslongtok},%
6098       long={\the\glslongtok},%
6099       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6100       text={\noexpand\@glo@short},%
6101       textaccess={\noexpand\@glo@shortaccess},%
6102       plural={\noexpand\@glo@shortpl},%
6103       first={\the\glslongtok},%
6104       firstaccess=\relax,
6105       firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6106       description={\noexpand\@glo@first},%
6107       descriptionplural={\noexpand\@glo@firstplural},%
6108       symbol={\the\glsshorttok},%
6109       symbolaccess={\the\glslongtok},%
6110       symbolplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6111       \the\glskeylisttok
6112     }%
6113   }%
6114   \do@newglossaryentry
6115 }

```

The following are kept for compatibility with versions before 3.0:

```

\glsshortaccesskey
6116 \newcommand*{\glsshortaccesskey}{\glsshortkey access}%

\glsshortpluralaccesskey
6117 \newcommand*{\glsshortpluralaccesskey}{\glsshortpluralkey access}%

\glslongaccesskey
6118 \newcommand*{\glslongaccesskey}{\glslongkey access}%

\glslongpluralaccesskey
6119 \newcommand*{\glslongpluralaccesskey}{\glslongpluralkey access}%

```

5.5 Debugging Commands

```

\showglongnameaccess
6120 \newcommand*{\showglongnameaccess}[1]{%
6121 \expandafter\show\csname glo@#1@textaccess\endcsname
6122 }

\showgлотextaccess
6123 \newcommand*{\showgлотextaccess}[1]{%
6124 \expandafter\show\csname glo@#1@textaccess\endcsname
6125 }

\showglopluralaccess
6126 \newcommand*{\showglopluralaccess}[1]{%
6127 \expandafter\show\csname glo@#1@pluralaccess\endcsname
6128 }

\showglofirstaccess
6129 \newcommand*{\showglofirstaccess}[1]{%
6130 \expandafter\show\csname glo@#1@firstaccess\endcsname
6131 }

\showglofirstpluralaccess
6132 \newcommand*{\showglofirstpluralaccess}[1]{%
6133 \expandafter\show\csname glo@#1@firstpluralaccess\endcsname
6134 }

\showglosymbolaccess
6135 \newcommand*{\showglosymbolaccess}[1]{%
6136 \expandafter\show\csname glo@#1@symbolaccess\endcsname
6137 }

\showglosymbolpluralaccess
6138 \newcommand*{\showglosymbolpluralaccess}[1]{%
6139 \expandafter\show\csname glo@#1@symbolpluralaccess\endcsname
6140 }

```

```

\showglodescaccess
6141 \newcommand*\showglodescaccess}[1]{%
6142 \expandafter\show\csname glo@#1@descaccess\endcsname
6143 }

\showglodescpluralaccess
6144 \newcommand*\showglodescpluralaccess}[1]{%
6145 \expandafter\show\csname glo@#1@descpluralaccess\endcsname
6146 }

\showgloshortaccess
6147 \newcommand*\showgloshortaccess}[1]{%
6148 \expandafter\show\csname glo@#1@shortaccess\endcsname
6149 }

\showgloshortpluralaccess
6150 \newcommand*\showgloshortpluralaccess}[1]{%
6151 \expandafter\show\csname glo@#1@shortpluralaccess\endcsname
6152 }

\showglolongaccess
6153 \newcommand*\showglolongaccess}[1]{%
6154 \expandafter\show\csname glo@#1@longaccess\endcsname
6155 }

\showglolongpluralaccess
6156 \newcommand*\showglolongpluralaccess}[1]{%
6157 \expandafter\show\csname glo@#1@longpluralaccess\endcsname
6158 }

```

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.

```

6159 \NeedsTeXFormat{LaTeX2e}
6160 \ProvidesPackage{glossaries-babel}[2009/04/16 v1.2 (NLCT)]

English:
6161 \@ifundefined{captionsenglish}{-}{%
6162 \addto\captionsenglish{%
6163 \renewcommand*\glossaryname}{Glossary}%
6164 \renewcommand*\acronymname}{Acronyms}%
6165 \renewcommand*\entryname}{Notation}%
6166 \renewcommand*\descriptionname}{Description}%

```

```

6167 \renewcommand*\symbolname}{Symbol}%
6168 \renewcommand*\pagelistname}{Page List}%
6169 \renewcommand*\glssymbolsgroupname}{Symbols}%
6170 \renewcommand*\glsnumbersgroupname}{Numbers}%
6171 }%
6172 }
6173 \@ifundefined{captionsamerican}{}{%
6174 \addto\captionsamerican{%
6175 \renewcommand*\glossaryname}{Glossary}%
6176 \renewcommand*\acronymname}{Acronyms}%
6177 \renewcommand*\entryname}{Notation}%
6178 \renewcommand*\descriptionname}{Description}%
6179 \renewcommand*\symbolname}{Symbol}%
6180 \renewcommand*\pagelistname}{Page List}%
6181 \renewcommand*\glssymbolsgroupname}{Symbols}%
6182 \renewcommand*\glsnumbersgroupname}{Numbers}%
6183 }%
6184 }
6185 \@ifundefined{captionsaustralian}{}{%
6186 \addto\captionsaustralian{%
6187 \renewcommand*\glossaryname}{Glossary}%
6188 \renewcommand*\acronymname}{Acronyms}%
6189 \renewcommand*\entryname}{Notation}%
6190 \renewcommand*\descriptionname}{Description}%
6191 \renewcommand*\symbolname}{Symbol}%
6192 \renewcommand*\pagelistname}{Page List}%
6193 \renewcommand*\glssymbolsgroupname}{Symbols}%
6194 \renewcommand*\glsnumbersgroupname}{Numbers}%
6195 }%
6196 }
6197 \@ifundefined{captionsbritish}{}{%
6198 \addto\captionsbritish{%
6199 \renewcommand*\glossaryname}{Glossary}%
6200 \renewcommand*\acronymname}{Acronyms}%
6201 \renewcommand*\entryname}{Notation}%
6202 \renewcommand*\descriptionname}{Description}%
6203 \renewcommand*\symbolname}{Symbol}%
6204 \renewcommand*\pagelistname}{Page List}%
6205 \renewcommand*\glssymbolsgroupname}{Symbols}%
6206 \renewcommand*\glsnumbersgroupname}{Numbers}%
6207 }%
6208 \@ifundefined{captionscanadian}{}{%
6209 \addto\captionscanadian{%
6210 \renewcommand*\glossaryname}{Glossary}%
6211 \renewcommand*\acronymname}{Acronyms}%
6212 \renewcommand*\entryname}{Notation}%
6213 \renewcommand*\descriptionname}{Description}%
6214 \renewcommand*\symbolname}{Symbol}%
6215 \renewcommand*\pagelistname}{Page List}%
6216 \renewcommand*\glssymbolsgroupname}{Symbols}%

```

```

6217 \renewcommand*\glsnumbersgroupname}{Numbers}%
6218 }%
6219 }
6220 \@ifundefined{captionsnewzealand}{}{%
6221 \addto\captionsnewzealand{%
6222 \renewcommand*\glossaryname}{Glossary}%
6223 \renewcommand*\acronymname}{Acronyms}%
6224 \renewcommand*\entryname}{Notation}%
6225 \renewcommand*\descriptionname}{Description}%
6226 \renewcommand*\symbolname}{Symbol}%
6227 \renewcommand*\pagelistname}{Page List}%
6228 \renewcommand*\glssymbolsgroupname}{Symbols}%
6229 \renewcommand*\glsnumbersgroupname}{Numbers}%
6230 }%
6231 }
6232 \@ifundefined{captionsUKenglish}{}{%
6233 \addto\captionsUKenglish{%
6234 \renewcommand*\glossaryname}{Glossary}%
6235 \renewcommand*\acronymname}{Acronyms}%
6236 \renewcommand*\entryname}{Notation}%
6237 \renewcommand*\descriptionname}{Description}%
6238 \renewcommand*\symbolname}{Symbol}%
6239 \renewcommand*\pagelistname}{Page List}%
6240 \renewcommand*\glssymbolsgroupname}{Symbols}%
6241 \renewcommand*\glsnumbersgroupname}{Numbers}%
6242 }%
6243 }
6244 \@ifundefined{captionsUSenglish}{}{%
6245 \addto\captionsUSenglish{%
6246 \renewcommand*\glossaryname}{Glossary}%
6247 \renewcommand*\acronymname}{Acronyms}%
6248 \renewcommand*\entryname}{Notation}%
6249 \renewcommand*\descriptionname}{Description}%
6250 \renewcommand*\symbolname}{Symbol}%
6251 \renewcommand*\pagelistname}{Page List}%
6252 \renewcommand*\glssymbolsgroupname}{Symbols}%
6253 \renewcommand*\glsnumbersgroupname}{Numbers}%
6254 }%
6255 }

```

German (quite a few variations were suggested for German; I settled on the following):

```

6256 \@ifundefined{captionsgerman}{}{%
6257 \addto\captionsgerman{%
6258 \renewcommand*\glossaryname}{Glossar}%
6259 \renewcommand*\acronymname}{Akronyme}%
6260 \renewcommand*\entryname}{Bezeichnung}%
6261 \renewcommand*\descriptionname}{Beschreibung}%
6262 \renewcommand*\symbolname}{Symbol}%
6263 \renewcommand*\pagelistname}{Seiten}%

```

```

6264 \renewcommand*{\glssymbolsgroupname}{Symbole}%
6265 \renewcommand*{\glsnumbersgroupname}{Zahlen}}
6266 }

```

ngerman is identical to German:

```

6267 \@ifundefined{captionsngerman}{-}{%
6268 \addto\captionsngerman{%
6269 \renewcommand*{\glossaryname}{Glossar}%
6270 \renewcommand*{\acronymname}{Akronyme}%
6271 \renewcommand*{\entryname}{Bezeichnung}%
6272 \renewcommand*{\descriptionname}{Beschreibung}%
6273 \renewcommand*{\symbolname}{Symbol}%
6274 \renewcommand*{\pagelistname}{Seiten}%
6275 \renewcommand*{\glssymbolsgroupname}{Symbole}%
6276 \renewcommand*{\glsnumbersgroupname}{Zahlen}}
6277 }

```

Italian:

```

6278 \@ifundefined{captionsitalian}{-}{%
6279 \addto\captionsitalian{%
6280 \renewcommand*{\glossaryname}{Glossario}%
6281 \renewcommand*{\acronymname}{Acronimi}%
6282 \renewcommand*{\entryname}{Nomenclatura}%
6283 \renewcommand*{\descriptionname}{Descrizione}%
6284 \renewcommand*{\symbolname}{Simbolo}%
6285 \renewcommand*{\pagelistname}{Elenco delle pagine}%
6286 \renewcommand*{\glssymbolsgroupname}{Simboli}%
6287 \renewcommand*{\glsnumbersgroupname}{Numeri}}
6288 }

```

Dutch:

```

6289 \@ifundefined{captionsdutch}{-}{%
6290 \addto\captionsdutch{%
6291 \renewcommand*{\glossaryname}{Woordenlijst}%
6292 \renewcommand*{\acronymname}{Acroniemen}%
6293 \renewcommand*{\entryname}{Benaming}%
6294 \renewcommand*{\descriptionname}{Beschrijving}%
6295 \renewcommand*{\symbolname}{Symbool}%
6296 \renewcommand*{\pagelistname}{Pagina's}%
6297 \renewcommand*{\glssymbolsgroupname}{Symbolen}%
6298 \renewcommand*{\glsnumbersgroupname}{Cijfers}}
6299 }

```

Spanish:

```

6300 \@ifundefined{captionsspanish}{-}{%
6301 \addto\captionsspanish{%
6302 \renewcommand*{\glossaryname}{Glosario}%
6303 \renewcommand*{\acronymname}{Siglas}%
6304 \renewcommand*{\entryname}{Entrada}%
6305 \renewcommand*{\descriptionname}{Descripci\`on}%
6306 \renewcommand*{\symbolname}{S\`{i}mbolo}%

```

```

6307 \renewcommand*\pagelistname}{Lista de p'aginas}%
6308 \renewcommand*\glssymbolsgroupname}{S'\{i\}mbolos}%
6309 \renewcommand*\glsnumbersgroupname}{N'\umeros}}
6310 }

```

French:

```

6311 \@ifundefined{captionsfrench}{-}{%
6312 \addto\captionsfrench{%
6313 \renewcommand*\glossaryname}{Glossaire}%
6314 \renewcommand*\acronymname}{Acronymes}%
6315 \renewcommand*\entryname}{Terme}%
6316 \renewcommand*\descriptionname}{Description}%
6317 \renewcommand*\symbolname}{Symbole}%
6318 \renewcommand*\pagelistname}{Pages}%
6319 \renewcommand*\glssymbolsgroupname}{Symboles}%
6320 \renewcommand*\glsnumbersgroupname}{Nombres}}
6321 }

```

```

6322 \@ifundefined{captionsfrenchb}{-}{%
6323 \addto\captionsfrenchb{%
6324 \renewcommand*\glossaryname}{Glossaire}%
6325 \renewcommand*\acronymname}{Acronymes}%
6326 \renewcommand*\entryname}{Terme}%
6327 \renewcommand*\descriptionname}{Description}%
6328 \renewcommand*\symbolname}{Symbole}%
6329 \renewcommand*\pagelistname}{Pages}%
6330 \renewcommand*\glssymbolsgroupname}{Symboles}%
6331 \renewcommand*\glsnumbersgroupname}{Nombres}}
6332 }

```

```

6333 \@ifundefined{captionsfrançais}{-}{%
6334 \addto\captionsfrançais{%
6335 \renewcommand*\glossaryname}{Glossaire}%
6336 \renewcommand*\acronymname}{Acronymes}%
6337 \renewcommand*\entryname}{Terme}%
6338 \renewcommand*\descriptionname}{Description}%
6339 \renewcommand*\symbolname}{Symbole}%
6340 \renewcommand*\pagelistname}{Pages}%
6341 \renewcommand*\glssymbolsgroupname}{Symboles}%
6342 \renewcommand*\glsnumbersgroupname}{Nombres}}
6343 }

```

Danish:

```

6344 \@ifundefined{captionsdanish}{-}{%
6345 \addto\captionsdanish{%
6346 \renewcommand*\glossaryname}{Ordliste}%
6347 \renewcommand*\acronymname}{Akronymer}%
6348 \renewcommand*\entryname}{Symbolforklaring}%
6349 \renewcommand*\descriptionname}{Beskrivelse}%
6350 \renewcommand*\symbolname}{Symbol}%
6351 \renewcommand*\pagelistname}{Side}%
6352 \renewcommand*\glssymbolsgroupname}{Symboler}%
6353 \renewcommand*\glsnumbersgroupname}{Tal}}

```

```

6354 }
      Irish:
6355 \@ifundefined{captionssirish}{-}{%
6356   \addto\captionssirish{%
6357     \renewcommand*\glossaryname{Gluais}%
6358     \renewcommand*\acronymname{Acrainmneacha}%
      wasn't sure whether to go for Nóta (Note), Ciall ('Meaning', 'sense') or Brí ('Mean-
      ing'). In the end I chose Ciall.
6359     \renewcommand*\entryname{Ciall}%
6360     \renewcommand*\descriptionname{Tuaisc}%
      Again, not sure whether to use Comhartha/Comharthaí or Siombail/Siombaile,
      so have chosen the former.
6361     \renewcommand*\symbolname{Comhartha}%
6362     \renewcommand*\glssymbolsgroupname{Comhartha\'}{i}%
6363     \renewcommand*\pagelistname{Leathanaigh}%
6364     \renewcommand*\glsnumbersgroupname{Uimhreacha}}
6365 }
      Hungarian:
6366 \@ifundefined{captionsmagyar}{-}{%
6367   \addto\captionsmagyar{%
6368     \renewcommand*\glossaryname{Sz\'ojegy\'ek}%
6369     \renewcommand*\acronymname{Bet\H uszavak}%
6370     \renewcommand*\entryname{Kifejez\'es}%
6371     \renewcommand*\descriptionname{Magyar\'azat}%
6372     \renewcommand*\symbolname{Jel\'ol\'es}%
6373     \renewcommand*\pagelistname{Oldalsz\'am}%
6374     \renewcommand*\glssymbolsgroupname{Jelek}%
6375     \renewcommand*\glsnumbersgroupname{Sz\'amjegyek}%
6376   }
6377 }
6378 \@ifundefined{captionshungarian}{-}{%
6379   \addto\captionshungarian{%
6380     \renewcommand*\glossaryname{Sz\'ojegy\'ek}%
6381     \renewcommand*\acronymname{Bet\H uszavak}%
6382     \renewcommand*\entryname{Kifejez\'es}%
6383     \renewcommand*\descriptionname{Magyar\'azat}%
6384     \renewcommand*\symbolname{Jel\'ol\'es}%
6385     \renewcommand*\pagelistname{Oldalsz\'am}%
6386     \renewcommand*\glssymbolsgroupname{Jelek}%
6387     \renewcommand*\glsnumbersgroupname{Sz\'amjegyek}%
6388   }
6389 }
      Polish
6390 \@ifundefined{captionsspolish}{-}{%
6391   \addto\captionsspolish{%
6392     \renewcommand*\glossaryname{S\{l\}ownik termin\'}{ow}%
6393     \renewcommand*\acronymname{Skr\'}{ot}%

```

```

6394 \renewcommand*\entryname}{Termin}%
6395 \renewcommand*\descriptionname}{Opis}%
6396 \renewcommand*\symbolname}{Symbol}%
6397 \renewcommand*\pagelistname}{Strony}%
6398 \renewcommand*\glssymbolsgroupname}{Symbole}%
6399 \renewcommand*\glsnumbersgroupname}{Liczby}}
6400 }

```

Brazilian

```

6401 \@ifundefined{captionsbrazil}{}{%
6402 \addto\captionsbrazil{%
6403 \renewcommand*\glossaryname}{Gloss\`ario}%
6404 \renewcommand*\acronymname}{Siglas}%
6405 \renewcommand*\entryname}{Nota\c c\~ao}%
6406 \renewcommand*\descriptionname}{Descri\c c\~ao}%
6407 \renewcommand*\symbolname}{S\`imbolo}%
6408 \renewcommand*\pagelistname}{Lista de P\`aginas}%
6409 \renewcommand*\glssymbolsgroupname}{S\`imbolos}%
6410 \renewcommand*\glsnumbersgroupname}{N\`umeros}%
6411 }%
6412 }

```

6.2 Polyglossia Captions

```

6413 \NeedsTeXFormat{LaTeX2e}
6414 \ProvidesPackage{glossaries-polyglossia}[2009/11/09 v1.0 (NLCT)]

```

English:

```

6415 \@ifundefined{captionseenglish}{}{%
6416 \expandafter\toks@\expandafter{\captionseenglish
6417 \renewcommand*\glossaryname}{\textenglish{Glossary}}%
6418 \renewcommand*\acronymname}{\textenglish{Acronyms}}%
6419 \renewcommand*\entryname}{\textenglish{Notation}}%
6420 \renewcommand*\descriptionname}{\textenglish{Description}}%
6421 \renewcommand*\symbolname}{\textenglish{Symbol}}%
6422 \renewcommand*\pagelistname}{\textenglish{Page List}}%
6423 \renewcommand*\glssymbolsgroupname}{\textenglish{Symbols}}%
6424 \renewcommand*\glsnumbersgroupname}{\textenglish{Numbers}}%
6425 }%
6426 \edef\captionseenglish{\the\toks@}%
6427 }

```

German:

```

6428 \@ifundefined{captionsgerman}{}{%
6429 \expandafter\toks@\expandafter{\captionsgerman
6430 \renewcommand*\glossaryname}{\textgerman{Glossar}}%
6431 \renewcommand*\acronymname}{\textgerman{Akronyme}}%
6432 \renewcommand*\entryname}{\textgerman{Bezeichnung}}%
6433 \renewcommand*\descriptionname}{\textgerman{Beschreibung}}%
6434 \renewcommand*\symbolname}{\textgerman{Symbol}}%
6435 \renewcommand*\pagelistname}{\textgerman{Seiten}}%

```

```

6436 \renewcommand*{\glssymbolsgroupname}{\textgerman{Symbole}}%
6437 \renewcommand*{\glsnumbersgroupname}{\textgerman{Zahlen}}%
6438 }%
6439 \edef\captionsgerman{\the\toks@}%
6440 }

Italian:
6441 \@ifundefined{captionsitalian}{}{%
6442 \expandafter\toks@\expandafter{\captionsitalian
6443 \renewcommand*{\glossaryname}{\textitalian{Glossario}}%
6444 \renewcommand*{\acronymname}{\textitalian{Acronimi}}%
6445 \renewcommand*{\entryname}{\textitalian{Nomenclatura}}%
6446 \renewcommand*{\descriptionname}{\textitalian{Descrizione}}%
6447 \renewcommand*{\symbolname}{\textitalian{Simbolo}}%
6448 \renewcommand*{\pagelistname}{\textitalian{Elenco delle pagine}}%
6449 \renewcommand*{\glssymbolsgroupname}{\textitalian{Simboli}}%
6450 \renewcommand*{\glsnumbersgroupname}{\textitalian{Numeri}}%
6451 }%
6452 \edef\captionsspanish{\the\toks@}%
6453 }

Dutch:
6454 \@ifundefined{captionsdutch}{}{%
6455 \expandafter\toks@\expandafter{\captionsdutch
6456 \renewcommand*{\glossaryname}{\textdutch{Woordenlijst}}%
6457 \renewcommand*{\acronymname}{\textdutch{Acroniemen}}%
6458 \renewcommand*{\entryname}{\textdutch{Benaming}}%
6459 \renewcommand*{\descriptionname}{\textdutch{Beschrijving}}%
6460 \renewcommand*{\symbolname}{\textdutch{Symbool}}%
6461 \renewcommand*{\pagelistname}{\textdutch{Pagina's}}%
6462 \renewcommand*{\glssymbolsgroupname}{\textdutch{Symbolen}}%
6463 \renewcommand*{\glsnumbersgroupname}{\textdutch{Cijfers}}%
6464 }%
6465 \edef\captionsdutch{\the\toks@}%
6466 }

Spanish:
6467 \@ifundefined{captionsspanish}{}{%
6468 \expandafter\toks@\expandafter{\captionsspanish
6469 \renewcommand*{\glossaryname}{\textspanish{Glosario}}%
6470 \renewcommand*{\acronymname}{\textspanish{Siglas}}%
6471 \renewcommand*{\entryname}{\textspanish{Entrada}}%
6472 \renewcommand*{\descriptionname}{\textspanish{Descripci'on}}%
6473 \renewcommand*{\symbolname}{\textspanish{S'\i mbolo}}%
6474 \renewcommand*{\pagelistname}{\textspanish{Lista de p' aginas}}%
6475 \renewcommand*{\glssymbolsgroupname}{\textspanish{S'\i mbolos}}%
6476 \renewcommand*{\glsnumbersgroupname}{\textspanish{N' umeros}}%
6477 }%
6478 \edef\captionsspanish{\the\toks@}%
6479 }

French:

```

```

6480 \@ifundefined{captionsfrench}{}{%
6481   \expandafter\toks@\expandafter{\captionsfrench
6482     \renewcommand*\glossaryname{\textfrench{Glossaire}}%
6483     \renewcommand*\acronymname{\textfrench{Acronymes}}%
6484     \renewcommand*\entryname{\textfrench{Terme}}%
6485     \renewcommand*\descriptionname{\textfrench{Description}}%
6486     \renewcommand*\symbolname{\textfrench{Symbole}}%
6487     \renewcommand*\pagelistname{\textfrench{Pages}}%
6488     \renewcommand*\glssymbolsgroupname{\textfrench{Symboles}}%
6489     \renewcommand*\glsnumbersgroupname{\textfrench{Nombres}}%
6490   }%
6491   \edef\captionsfrench{\the\toks@}%
6492 }

```

Danish:

```

6493 \@ifundefined{captionsdanish}{}{%
6494   \expandafter\toks@\expandafter{\captionsdanish
6495     \renewcommand*\glossaryname{\textdanish{Ordliste}}%
6496     \renewcommand*\acronymname{\textdanish{Akronymer}}%
6497     \renewcommand*\entryname{\textdanish{Symbolforklaring}}%
6498     \renewcommand*\descriptionname{\textdanish{Beskrivelse}}%
6499     \renewcommand*\symbolname{\textdanish{Symbol}}%
6500     \renewcommand*\pagelistname{\textdanish{Side}}%
6501     \renewcommand*\glssymbolsgroupname{\textdanish{Symboler}}%
6502     \renewcommand*\glsnumbersgroupname{\textdanish{Tal}}%
6503   }%
6504   \edef\captionsdanish{\the\toks@}%
6505 }

```

Irish:

```

6506 \@ifundefined{captionsirish}{}{%
6507   \expandafter\toks@\expandafter{\captionsirish
6508     \renewcommand*\glossaryname{\textirish{Gluais}}%
6509     \renewcommand*\acronymname{\textirish{Acraimneacha}}%
6510     \renewcommand*\entryname{\textirish{Ciall}}%
6511     \renewcommand*\descriptionname{\textirish{Tuaisc}}%
6512     \renewcommand*\symbolname{\textirish{Comhartha}}%
6513     \renewcommand*\glssymbolsgroupname{\textirish{Comhartha\'}{i}}%
6514     \renewcommand*\pagelistname{\textirish{Leathanaigh}}%
6515     \renewcommand*\glsnumbersgroupname{\textirish{Uimhreacha}}%
6516   }%
6517   \edef\captionsirish{\the\toks@}%
6518 }

```

Hungarian:

```

6519 \@ifundefined{captionsmagyar}{}{%
6520   \expandafter\toks@\expandafter{\captionsmagyar
6521     \renewcommand*\glossaryname{\textmagyar{Sz\'}{ojegyz\'}{ek}}%
6522     \renewcommand*\acronymname{\textmagyar{Bet\'}{H uszavak}}%
6523     \renewcommand*\entryname{\textmagyar{Kifejez\'}{es}}%
6524     \renewcommand*\descriptionname{\textmagyar{Magyar\'}{azat}}%
6525     \renewcommand*\symbolname{\textmagyar{Jel\'}{ol\'}{es}}%

```

```

6526 \renewcommand*\pagelistname{\textmagyar{Oldalsz\ 'am}}%
6527 \renewcommand*\glssymbolsgroupname{\textmagyar{Jelek}}%
6528 \renewcommand*\glsnumbersgroupname{\textmagyar{Sz\ 'amjegyek}}%
6529 }%
6530 \edef\captionsmagyar{\the\toks@}%
6531 }

Polish
6532 \@ifundefined{captionspolish}{}{%
6533 \expandafter\toks@\expandafter{\captionspolish
6534 \renewcommand*\glossaryname{\textpolish{S{\l}ownik termin\ 'ow}}%
6535 \renewcommand*\acronymname{\textpolish{Skr\ 'ot}}%
6536 \renewcommand*\entryname{\textpolish{Termin}}%
6537 \renewcommand*\descriptionname{\textpolish{Opis}}%
6538 \renewcommand*\symbolname{\textpolish{Symbol}}%
6539 \renewcommand*\pagelistname{\textpolish{Strony}}%
6540 \renewcommand*\glssymbolsgroupname{\textpolish{Symbole}}%
6541 \renewcommand*\glsnumbersgroupname{\textpolish{Liczby}}%
6542 }%
6543 \edef\captionspolish{\the\toks@}%
6544 }

Portugues
6545 \@ifundefined{captionportuges}{}{%
6546 \expandafter\toks@\expandafter{\captionportuges
6547 \renewcommand*\glossaryname{\textportuges{Gloss\ 'ario}}%
6548 \renewcommand*\acronymname{\textportuges{Siglas}}%
6549 \renewcommand*\entryname{\textportuges{Nota\c c\~ao}}%
6550 \renewcommand*\descriptionname{\textportuges{Descri\c c\~ao}}%
6551 \renewcommand*\symbolname{\textportuges{S\ 'imbolo}}%
6552 \renewcommand*\pagelistname{\textportuges{Lista de P\ 'aginas}}%
6553 \renewcommand*\glssymbolsgroupname{\textportuges{S\ 'imbolos}}%
6554 \renewcommand*\glsnumbersgroupname{\textportuges{N\ 'umeros}}%
6555 }%
6556 \edef\captionportuges{\the\toks@}%
6557 }

```

6.3 Brazilian Dictionary

This is a dictionary file provided by Thiago de Melo for use with the package.

```

6558 \ProvidesDictionary{glossaries-dictionary}{Brazilian}

Provide Brazilian translations:
6559 \providetranslation{Glossary}{Gloss\ 'ario}
6560 \providetranslation{Acronyms}{Siglas}
6561 \providetranslation{Notation (glossaries)}{Nota\c c\~ao}
6562 \providetranslation{Description (glossaries)}{Descri\c c\~ao}
6563 \providetranslation{Symbol (glossaries)}{S\ 'imbolo}
6564 \providetranslation{Page List (glossaries)}{Lista de P\ 'aginas}
6565 \providetranslation{Symbols (glossaries)}{S\ 'imbolos}
6566 \providetranslation{Numbers (glossaries)}{N\ 'umeros}

```

6.4 Danish Dictionary

This is a dictionary file provided for use with the package.

```
6567 \ProvidesDictionary{glossaries-dictionary}{Danish}
```

Provide Danish translations:

```
6568 \providetranslation{Glossary}{Ordliste}
6569 \providetranslation{Acronyms}{Akronymer}
6570 \providetranslation{Notation (glossaries)}{Symbolforklaring}
6571 \providetranslation{Description (glossaries)}{Beskrivelse}
6572 \providetranslation{Symbol (glossaries)}{Symbol}
6573 \providetranslation{Page List (glossaries)}{Side}
6574 \providetranslation{Symbols (glossaries)}{Symboler}
6575 \providetranslation{Numbers (glossaries)}{Tal}
```

6.5 Dutch Dictionary

This is a dictionary file provided for use with the package.

```
6576 \ProvidesDictionary{glossaries-dictionary}{Dutch}
```

Provide Dutch translations:

```
6577 \providetranslation{Glossary}{Woordenlijst}
6578 \providetranslation{Acronyms}{Acroniemen}
6579 \providetranslation{Notation (glossaries)}{Benaming}
6580 \providetranslation{Description (glossaries)}{Beschrijving}
6581 \providetranslation{Symbol (glossaries)}{Symbool}
6582 \providetranslation{Page List (glossaries)}{Pagina's}
6583 \providetranslation{Symbols (glossaries)}{Symbolen}
6584 \providetranslation{Numbers (glossaries)}{Cijfers}
```

6.6 English Dictionary

This is a dictionary file provided for use with the package.

```
6585 \ProvidesDictionary{glossaries-dictionary}{English}
```

Provide English translations:

```
6586 \providetranslation{Glossary}{Glossary}
6587 \providetranslation{Acronyms}{Acronyms}
6588 \providetranslation{Notation (glossaries)}{Notation}
6589 \providetranslation{Description (glossaries)}{Description}
6590 \providetranslation{Symbol (glossaries)}{Symbol}
6591 \providetranslation{Page List (glossaries)}{Page List}
6592 \providetranslation{Symbols (glossaries)}{Symbols}
6593 \providetranslation{Numbers (glossaries)}{Numbers}
```

6.7 French Dictionary

This is a dictionary file provided for use with the package.

```
6594 \ProvidesDictionary{glossaries-dictionary}{French}
```

Provide French translations:

```
6595 \providetranslation{Glossary}{Glossaire}
6596 \providetranslation{Acronyms}{Acronymes}
6597 \providetranslation{Notation (glossaries)}{Termes}
6598 \providetranslation{Description (glossaries)}{Description}
6599 \providetranslation{Symbol (glossaries)}{Symbole}
6600 \providetranslation{Page List (glossaries)}{Pages}
6601 \providetranslation{Symbols (glossaries)}{Symboles}
6602 \providetranslation{Numbers (glossaries)}{Nombres}
```

6.8 German Dictionary

This is a dictionary file provided for use with the package.

```
6603 \ProvidesDictionary{glossaries-dictionary}{German}
```

Provide German translations (quite a few variations were suggested for German; I settled on the following):

```
6604 \providetranslation{Glossary}{Glossar}
6605 \providetranslation{Acronyms}{Akronyme}
6606 \providetranslation{Notation (glossaries)}{Bezeichnung}
6607 \providetranslation{Description (glossaries)}{Beschreibung}
6608 \providetranslation{Symbol (glossaries)}{Symbol}
6609 \providetranslation{Page List (glossaries)}{Seiten}
6610 \providetranslation{Symbols (glossaries)}{Symbole}
6611 \providetranslation{Numbers (glossaries)}{Zahlen}
```

6.9 Irish Dictionary

This is a dictionary file provided for use with the package.

```
6612 \ProvidesDictionary{glossaries-dictionary}{Irish}
```

Provide Irish translations:

```
6613 \providetranslation{Glossary}{Gluais}
6614 \providetranslation{Acronyms}{Acrainmneacha}
6615 \providetranslation{Notation (glossaries)}{Ciall}
6616 \providetranslation{Description (glossaries)}{Tuairisc}
6617 \providetranslation{Symbol (glossaries)}{Comhartha}
6618 \providetranslation{Page List (glossaries)}{Leathanaigh}
6619 \providetranslation{Symbols (glossaries)}{Comhartha\'}{i}
6620 \providetranslation{Numbers (glossaries)}{Uimhreacha}
```

6.10 Italian Dictionary

This is a dictionary file provided for use with the package.

```
6621 \ProvidesDictionary{glossaries-dictionary}{Italian}
```

Provide Italian translations:

```
6622 \providetranslation{Glossary}{Glossario}
6623 \providetranslation{Acronyms}{Acronimi}
```

```

6624 \providetranslation{Notation (glossaries)}{Nomenclatura}
6625 \providetranslation{Description (glossaries)}{Descrizione}
6626 \providetranslation{Symbol (glossaries)}{Simbolo}
6627 \providetranslation{Page List (glossaries)}{Elenco delle pagine}
6628 \providetranslation{Symbols (glossaries)}{Simboli}
6629 \providetranslation{Numbers (glossaries)}{Numeri}

```

6.11 Magyar Dictionary

This is a dictionary file provided for use with the package.

```
6630 \ProvidesDictionary{glossaries-dictionary}{Magyar}
```

Provide translations:

```

6631 \providetranslation{Glossary}{Sz\’ojegyz\’ek}
6632 \providetranslation{Acronyms}{Bet\H uszavak}
6633 \providetranslation{Notation (glossaries)}{Kifejez\’es}
6634 \providetranslation{Description (glossaries)}{Magyar\’azat}
6635 \providetranslation{Symbol (glossaries)}{Jel\’ol\’es}
6636 \providetranslation{Page List (glossaries)}{Oldalsz\’am}
6637 \providetranslation{Symbols (glossaries)}{Jelek}
6638 \providetranslation{Numbers (glossaries)}{Sz\’amjegyek}

```

6.12 Polish Dictionary

This is a dictionary file provided for use with the package.

```
6639 \ProvidesDictionary{glossaries-dictionary}{Polish}
```

Provide Polish translations:

```

6640 \providetranslation{Glossary}{S{\l}ownik termin\’ow}
6641 \providetranslation{Acronyms}{Skr\’ot}
6642 \providetranslation{Notation (glossaries)}{Termin}
6643 \providetranslation{Description (glossaries)}{Opis}
6644 \providetranslation{Symbol (glossaries)}{Symbol}
6645 \providetranslation{Page List (glossaries)}{Strony}
6646 \providetranslation{Symbols (glossaries)}{Symbole}
6647 \providetranslation{Numbers (glossaries)}{Liczby}

```

6.13 Serbian Dictionary

This dictionary was provided by Zoran Filipovic.

```

6648 \ProvidesDictionary{glossaries-dictionary}{Serbian}
6649 \providetranslation{Glossary}{Mali re\’v cnik}
6650 \providetranslation{Acronyms}{Skra\’ cenice}
6651 \providetranslation{Notation (glossaries)}{Oznaka}
6652 \providetranslation{Description (glossaries)}{Opis}
6653 \providetranslation{Symbol (glossaries)}{Simbol}
6654 \providetranslation{Page List (glossaries)}{Stranica}
6655 \providetranslation{Symbols (glossaries)}{Simboli}
6656 \providetranslation{Numbers (glossaries)}{Brojevi}

```

6.14 Spanish Dictionary

This is a dictionary file provided for use with the package.

```
6657 \ProvidesDictionary{glossaries-dictionary}{Spanish}
```

Provide Spanish translations:

```
6658 \providetranslation{Glossary}{Glosario}
6659 \providetranslation{Acronyms}{Siglas}
6660 \providetranslation{Notation (glossaries)}{Entrada}
6661 \providetranslation{Description (glossaries)}{Descripci'on}
6662 \providetranslation{Symbol (glossaries)}{S'\{i}mbolo}
6663 \providetranslation{Page List (glossaries)}{Lista de p'aginas}
6664 \providetranslation{Symbols (glossaries)}{S'\{i}mbolos}
6665 \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 | 4 |
| \@glossaryseclabel | 4 |
| \@glossarysecstar | 4 |
| \@ACRlong | 214 |
| \@ACRshort | 213 |
| \@Acrlong | 214 |
| \@Acrshort | 213 |
| \@GLS@ | 66 , 209 |
| \@GLSpl | 69 |
| \@GLSpl@ | 212 |
| \@Gls@ | 64 , 208 |
| \@Glspl@ | 68 , 211 |
| \@acrlong | 214 |
| \@acrshort | 213 |
| \@addtoacronymlists | 10 |
| \@delimN | 130 |
| \@delimR | 130 |
| \@disable@onlypremakeg | 16 |
| \@disable@premakecs | 16 |
| \@disabled@glssaddxdycounters | 26 |
| \@do@seeglossary | 118 |
| \@do@wrglossary | 116 , 194 |
| \@glo@seeautonumberlist | 5 |
| \@glo@storeentry | 47 |
| \@glo@types | 36 |
| \@glossary | 116 |
| \@glossary@default@style | 5 |
| \@glossaryentryfield | 46 , 215 |
| \@glossarysection | 23 |
| \@glossarysubentryfield | 47 , 215 |
| \@gls | 63 |
| \@gls@ | 63 , 207 |
| \@gls@@link | 53 |
| \@gls@addpredefinedattributes | 27 |
| \@gls@checkactual | 60 |
| \@gls@checkbar | 59 |
| \@gls@checkesactual | 57 |
| \@gls@checkeschar | 58 |
| \@gls@checkesclevel | 58 |
| \@gls@checkescquote | 57 |
| \@gls@checklevel | 59 |
| \@gls@checkmkidxchars | 56 |
| \@gls@checkquote | 56 |
| \@gls@codepage | 33 |
| \@gls@counterwithin | 6 |
| \@gls@escbsdq | 55 |
| \@gls@fixbraces | 118 |
| \@gls@getcounter | 38 |
| \@gls@getcounterprefix | 117 |
| \@gls@hypergroup | 158 |
| \@gls@ifinlist | 25 |
| \@gls@link | 53 |
| \@gls@loadlist | 6 |
| \@gls@loadlong | 5 |
| \@gls@loadsuper | 5 |
| \@gls@loadtree | 6 |
| \@gls@noaccess | 201 |
| \@gls@onlypremakeg | 16 |
| \@gls@pl@ | 210 |
| \@gls@renewglossary | 116 |
| \@gls@sanitizedesc | 11 |
| \@gls@sanitizename | 11 |
| \@gls@sanitizesymbol | 12 |
| \@gls@saveentrycounter | 53 |
| \@gls@setcounter | 37 |
| \@gls@setupsort@def | 7 |
| \@gls@setupsort@standard | 7 |
| \@gls@setupsort@use | 8 |
| \@gls@tmpb | 56 |
| \@gls@toc | 24 |
| \@gls@updatechecked | 56 |
| \@gls@xdy@Lclass@Alpha-page-numbers | 29 |
| \@gls@xdy@Lclass@Appendix-page-numbers | 29 |
| \@gls@xdy@Lclass@Roman-page-numbers | 28 |
| \@gls@xdy@Lclass@alpha-page-numbers | 29 |
| \@gls@xdy@Lclass@arabic-page-numbers | 29 |
| \@gls@xdy@Lclass@arabic-section-numbers | 29 |
| \@gls@xdy@Lclass@roman-page-numbers | 28 |

| | | | |
|---|--|--|---|
| <code>\@gls@xdy@locationlist</code> | 28 | <code>\accsuppglossaryentryfield</code> | 215 |
| <code>\@gls@xdycheckbackslash</code> | 61 | <code>\accsuppglossarysubentryfield</code> | 215 |
| <code>\@gls@xdycheckquote</code> | 60 | <code>\Acf</code> | 149 |
| <code>\@glsAlphacompositor</code> | 20 , 29 | <code>\acf</code> | 149 |
| <code>\@glsacronymlists</code> | 9 | <code>\Acfp</code> | 149 |
| <code>\@glsdefaultplural</code> | 42 | <code>\acfp</code> | 149 |
| <code>\@glsdefaultsort</code> | 42 | <code>\Acl</code> | 149 |
| <code>\@glsdisp</code> | 70 | <code>\acl</code> | 148 |
| <code>\@glsfirstletter</code> | 106 | <code>\Aclp</code> | 149 |
| <code>\@glshypernumber</code> | 129 | <code>\aclp</code> | 148 |
| <code>\@glslink</code> | 62 | <code>\Acp</code> | 149 |
| <code>\@glsminrange</code> | 107 | <code>\acp</code> | 149 |
| <code>\@glsnextpages</code> | 123 | <code>\acrfootnote</code> | 137 |
| <code>\@glsnoname</code> | 42 | <code>\ACRfull</code> | 134 |
| <code>\@glsnonextpages</code> | 123 | <code>\Acrfull</code> | 134 |
| <code>\@glsopenfile</code> | 113 | <code>\acrfull</code> | 133 |
| <code>\@glsorder</code> | 13 | <code>\acrfullformat</code> | 133 |
| <code>\@glspl@</code> | 66 | <code>\ACRfullpl</code> | 135 |
| <code>\@glstarget</code> | 62 | <code>\Acrfullpl</code> | 135 |
| <code>\@glswidestname</code> | 189 | <code>\acrfullpl</code> | 134 |
| <code>\@istfilename</code> | 20 | <code>\acrlinkfootnote</code> | 137 |
| <code>\@makeglossary</code> | 113 | <code>\acrlinkfullformat</code> | 133 |
| <code>\@newglossary</code> | 37 | <code>\ACRlong</code> | 98 |
| <code>\@newglossaryentryposthook</code> | 46 | <code>\Acrlong</code> | 97 |
| <code>\@newglossaryentryprehook</code> | 46 | <code>\acrlong</code> | 96 |
| <code>\@nopostdesc</code> | 19 | <code>\ACRlongpl</code> | 99 |
| <code>\@onlypremakeg</code> | 16 | <code>\Acrlongpl</code> | 99 |
| <code>\@p@glossarysection</code> | 23 | <code>\acrlongpl</code> | 98 |
| <code>\@set@glo@numformat</code> | 55 , 195 | <code>\acrnameformat</code> | 136 , 141 |
| <code>\@sgls</code> | 63 , 70 | <code>\acrnolinkfootnote</code> | 137 |
| <code>\@sgls@link</code> | 53 | <code>acronym (option)</code> | 9 |
| <code>\@wrglossary</code> | 116 | <code>\acronymfont</code> | 135 , 139 , 141 , 143 , 144 |
| <code>\@xdy@main@language</code> | 14 | <code>acronymlists (option)</code> | 11 |
| <code>\@xdyattributelist</code> | 25 | <code>\acronymname</code> | 16 |
| <code>\@xdyattributes</code> | 24 | <code>\acronymtype</code> | 9 , 131 , 132 |
| <code>\@xdy@language</code> | 33 | <code>\acrpluralsuffix</code> | 132 |
| <code>\@xdy@lettergroups</code> | 33 | <code>\ACRshort</code> | 94 |
| <code>\@xdy@locationclassorder</code> | 30 | <code>\Acrshort</code> | 93 |
| <code>\@xdy@locoref</code> | 25 | <code>\acrshort</code> | 93 |
| <code>\@xdy@requiredstyles</code> | 31 | <code>\ACRshortpl</code> | 96 |
| <code>\@xdy@sortrules</code> | 31 | <code>\Acrshortpl</code> | 95 |
| <code>\@xdy@useralphabets</code> | 27 | <code>\acrshortpl</code> | 95 |
| <code>\@xdy@userlocationdefs</code> | 29 | <code>\Acs</code> | 148 |
| <code>\@xdy@userlocationnames</code> | 29 | <code>\acs</code> | 148 |
| | | <code>\Acsp</code> | 148 |
| | A | <code>\acsp</code> | 148 |
| <code>\Ac</code> | 149 | <code>\addglossarytocaptions</code> | 17 |
| <code>\ac</code> | 149 | <code>\addto</code> | 17 |
| <code>access (key)</code> | 199 | <code>align (environment)</code> | 53 , 54 |
| <code>accsupp package</code> | 199 | | |

| | | | |
|--------------------------------|-------------------|--------------------------------------|---------------|
| glossaries-accsupp package .. | 46, 47, 199 | altlistgroup | 161 |
| \GlossariesWarning | 15 | altlisthypergroup | 161 |
| \GlossariesWarningNoLine | 15 | altlong4col | 166, 167, 171 |
| \glossary | 36, 113, 116, 128 | altlong4colborder | 167 |
| glossary counters: | | altlong4colheader | 167 |
| glossaryentry | 124 | altlong4colheaderborder | 167 |
| glossarysubentry | 124 | altlongragged4col | 171, 172 |
| glossary keys: | | altlongragged4colborder | 172 |
| access | 199 | altlongragged4colheader | 172 |
| counter | 40 | altlongragged4colheaderborder | 172 |
| description | 38 | altsuper4col | 178, 179, 183 |
| descriptionaccess | 200 | altsuper4colborder | 178 |
| descriptionplural | 38 | altsuper4colheader | 178 |
| descriptionpluralaccess | 200 | altsuper4colheaderborder | 179 |
| first | 39 | altsuperragged4col | 183, 184 |
| firstaccess | 200 | altsuperragged4colborder | 184 |
| firstplural | 39 | altsuperragged4colheader | 183 |
| firstpluralaccess | 200 | altsuperragged4colheaderborder | 184 |
| long | 41 | alttree | 189, 192 |
| longaccess | 201 | alttreegroup | 192 |
| longplural | 41 | alttreehypergroup | 192 |
| longpluralaccess | 201 | index | 185, 186 |
| name | 38 | indexgroup | 186 |
| nonumberlist | 40 | indexhypergroup | 186 |
| parent | 40 | list | 4, 159–161 |
| plural | 39 | listdotted | 161, 162 |
| pluralaccess | 200 | listgroup | 160 |
| see | 40 | listhypergroup | 160 |
| short | 41 | long | 162–164, 168 |
| shortaccess | 200 | long3col | 164, 165 |
| shortplural | 41 | long3colborder | 164 |
| shortpluralaccess | 200 | long3colheader | 165 |
| sort | 39 | long3colheaderborder | 165 |
| symbol | 39 | long4col | 165, 166 |
| symbolaccess | 200 | long4colborder | 166 |
| symbolplural | 40 | long4colheader | 166 |
| symbolpluralaccess | 200 | long4colheaderborder | 166 |
| text | 39 | longborder | 163 |
| textaccess | 200 | longheader | 163 |
| type | 40 | longheaderborder | 163 |
| user1 | 41 | longragged | 168, 169 |
| user2 | 41 | longragged3col | 169, 170 |
| user3 | 41 | longragged3colborder | 170 |
| user4 | 41 | longragged3colheader | 170 |
| user5 | 41 | longragged3colheaderborder | 171 |
| user6 | 41 | longraggedborder | 169 |
| glossary package | 1, 131 | longraggedheader | 169 |
| glossary styles: | | | |
| altlist | 160, 161 | | |

| | | | |
|----------------------------------|--------------------------|------------------------------------|---|
| longraggedheaderborder | 169 | \glossarystyle | 128, 150 |
| sublistdotted | 162 | glossarysubentry (counter) | 124 |
| super | 173–175, 181 | glossarysubentry counter | 6, 124, 125 |
| super3col | 175, 176 | \GLS | 65 |
| super3colborder | 175 | \Gls | 64, 67, 155 |
| super3colheader | 176 | \gls | 3, 39, 50, 51, 63, 65, 66, 71–73, 75, 76, 78, 79, 81, 82, 84, 85, 87, 88, 90, 91, 125 |
| super3colheaderborder | 176 | \gls@checkisacronymlist | 10 |
| super4col | 176–178 | \gls@codepage | 14 |
| super4colborder | 177 | \gls@docclearpage | 24 |
| super4colheader | 177 | \gls@hypergroup prerun | 158 |
| super4colheaderborder | 177 | \gls@level | 42 |
| superborder | 174 | \gls@suffixF | 21 |
| superheader | 174 | \gls@suffixFF | 21 |
| superheaderborder | 174 | \glsaccessdisplay | 207 |
| superragged | 180, 181 | \glsaccsupp | 204 |
| superragged3col | 181, 182 | \glsadd | 50, 105, 128 |
| superragged3colborder | 182 | \glsadd options | |
| superragged3colheader | 182 | counter | 105 |
| superragged3colheaderborder | 182 | format | 105, 129 |
| superraggedborder | 180 | \glsaddall | 50, 106 |
| superraggedheader | 181 | \glsaddall options | |
| superraggedheaderborder | 181 | types | 105, 106 |
| superraggedright3colheaderborder | 182 | \GlsAddLetterGroup | 34 |
| tree | 186–189 | \GlsAddSortRule | 31 |
| treegroup | 187 | \GlsAddXdyAlphabet | 27 |
| treehypergroup | 187 | \GlsAddXdyAttribute | 26, 193 |
| treenoname | 188, 189 | \GlsAddXdyCounters | 25, 193 |
| treenonamegroup | 189 | \GlsAddXdyLocation | 29, 194 |
| treenonamehypergroup | 189 | \GlsAddXdyStyle | 31 |
| glossary-hypernav package | 106 | \glsautoprefix | 4 |
| glossary-list package | 4, 6, 159 | \glsclearpage | 24 |
| glossary-long package | 5, 162, 171, 173 | \glsclosebrace | 106 |
| glossary-longragged package | 168 | \glscompositor | 20, 29 |
| glossary-super package | | \glscounter | 11, 37 |
| | 5, 6, 162, 173, 179, 183 | \glsdefaulttype | 9 |
| glossary-superragged package | 179 | \glsdefmain | 9 |
| glossary-tree package | 6, 185 | \GLSdesc | 79 |
| glossaryentry (counter) | 124 | \Glsdesc | 78 |
| glossaryentry counter | 6, 124, 125 | \glsdesc | 78, 79 |
| \glossaryentryfield | 39, 126, 128, 129 | \GLSdescplural | 80 |
| \glossaryentrynumber | 123 | \Glsdescplural | 80 |
| \glossaryentrynumbers | 5, 21, 120, 122 | \glsdescplural | 79, 80 |
| \glossaryheader | 126, 128 | \glsdescriptionaccessdisplay | 205 |
| \glossarymark | 22 | \glsdescriptionpluralaccessdisplay | 206 |
| \glossaryname | 16, 17 | | |
| \glossarypostamble | 22, 128 | \glsdescwidth | 162, 168, 173, 179 |
| \glossarypreamble | 22, 128 | \glsdisablehyper | 62 |
| \glossarysection | 4, 22, 36 | | |

| | | | |
|--|---|---|--|
| <code>\glsdisp</code> | 69 | <code>\Glsentrytext</code> | 101 |
| <code>\glsdisplay</code> | 11 , 38 , 39 , 50 , 51 , 63 | <code>\glsentrytext</code> | 101 , 119 |
| <code>\glsdisplayfirst</code> ... | 38 , 39 , 50 , 51 , 63 | <code>\glsentrytextaccess</code> | 202 |
| <code>\glsdoifexists</code> | 35 | <code>\glsentrytype</code> | 102 |
| <code>\glsdoifnoexists</code> | 35 | <code>\Glsentryuseri</code> | 103 |
| <code>\glsenablehyper</code> | 63 | <code>\glsentryuseri</code> | 102 |
| <code>\glsentryaccess</code> | 202 | <code>\Glsentryuserii</code> | 103 |
| <code>\glsentrycounterlabel</code> | 125 | <code>\glsentryuserii</code> | 103 |
| <code>\Glsentrydesc</code> | 101 | <code>\Glsentryuseriii</code> | 103 |
| <code>\glsentrydesc</code> | 11 , 100 | <code>\glsentryuseriii</code> | 103 |
| <code>\glsentrydescaccess</code> | 203 | <code>\Glsentryuseriv</code> | 103 |
| <code>\Glsentrydescplural</code> | 101 | <code>\glsentryuseriv</code> | 103 |
| <code>\glsentrydescplural</code> | 101 | <code>\Glsentryuserv</code> | 103 |
| <code>\glsentrydescpluralaccess</code> | 203 | <code>\glsentryuserv</code> | 103 |
| <code>\Glsentryfirst</code> | 102 | <code>\Glsentryuservi</code> | 103 |
| <code>\glsentryfirst</code> | 102 | <code>\glsentryuservi</code> | 103 |
| <code>\glsentryfirstaccess</code> | 202 | <code>\GLSfirst</code> | 73 |
| <code>\Glsentryfirstplural</code> | 102 | <code>\Glsfirst</code> | 73 |
| <code>\glsentryfirstplural</code> | 102 | <code>\glsfirst</code> | 72 |
| <code>\glsentryfirstpluralaccess</code> | 203 | <code>\glsfirstaccessdisplay</code> | 205 |
| <code>\Glsentryfull</code> | 104 | <code>\GLSfirstplural</code> | 76 |
| <code>\glsentryfull</code> | 104 | <code>\Glsfirstplural</code> | 75 |
| <code>\Glsentryfullpl</code> | 105 | <code>\glsfirstplural</code> | 75 , 76 |
| <code>\glsentryfullpl</code> | 105 | <code>\glsfirstpluralaccessdisplay</code> ... | 205 |
| <code>\glsentryitem</code> | 125 | <code>\glsgetgrouplabel</code> | 127 |
| <code>\Glsentrylong</code> | 104 | <code>\glsgetgrouptitle</code> | 106 , 127 |
| <code>\glsentrylong</code> | 104 | <code>\glsgroupheading</code> | 127 , 128 |
| <code>\glsentrylongaccess</code> | 203 | <code>\glsgroupskip</code> | 127 , 128 , 159 |
| <code>\Glsentrylongpl</code> | 104 | <code>\glshyperlink</code> | 105 |
| <code>\glsentrylongpl</code> | 104 | <code>\glshypernavsep</code> | 159 |
| <code>\glsentrylongpluralaccess</code> | 204 | <code>\glshypernumber</code> | 21 , 129 |
| <code>\Glsentryname</code> | 100 | <code>\glsIfListOfAcronyms</code> | 10 |
| <code>\glsentryname</code> | 100 , 119 | <code>\glskeylisttok</code> | 136 |
| <code>\Glsentryplural</code> | 101 | <code>\glslabeltok</code> | 136 |
| <code>\glsentryplural</code> | 101 | <code>\glslink</code> | 50 , 52 , 63 , 105 , 128 , 129 |
| <code>\glsentrypluralaccess</code> | 203 | <code>\glslink options</code> | |
| <code>\Glsentryshort</code> | 104 | counter | 52 , 63 , 155 |
| <code>\glsentryshort</code> | 104 | format | 52 , 63 , 129 |
| <code>\glsentryshortaccess</code> | 203 | hyper | 52 , 63 |
| <code>\Glsentryshortpl</code> | 104 | <code>\glslistdottedwidth</code> | 162 |
| <code>\glsentryshortpl</code> | 104 | <code>\glslocalreset</code> | 49 |
| <code>\glsentryshortpluralaccess</code> | 203 | <code>\glslocalresetall</code> | 49 |
| <code>\glsentrysort</code> | 102 | <code>\glslocalunset</code> | 49 |
| <code>\Glsentrysymbol</code> | 101 | <code>\glslocalunsetall</code> | 50 |
| <code>\glsentrysymbol</code> | 101 | <code>\glslocalunsetall</code> | 50 |
| <code>\glsentrysymbolaccess</code> | 203 | <code>\glslocalunsetting</code> | 206 |
| <code>\Glsentrysymbolplural</code> | 102 | <code>\glslocalunsettingkey</code> | 219 |
| <code>\glsentrysymbolplural</code> | 102 | <code>\glslocalunsettingkey</code> | 133 |
| <code>\glsentrysymbolpluralaccess</code> | 203 | <code>\glslocalunsettingkeydisplay</code> | 206 |
| | | <code>\glslocalunsettingkey</code> | 219 |

| | | | |
|---------------------------------------|------------------------------------|---|------------------------------|
| <code>\glslongpluralkey</code> | 133 | <code>\GlsSetXdyFirstLetterAfterDigits</code> | 107 |
| <code>\glslongtok</code> | 136 | <code>\GlsSetXdyLanguage</code> | 33 |
| <code>\glsmakefirsttuc</code> | 156 | <code>\GlsSetXdyLocationClassOrder</code> | 30 |
| <code>\GLSname</code> | 77 | <code>\GlsSetXdyMinRangeLength</code> | 107 |
| <code>\Glsname</code> | 77 | <code>\GlsSetXdyStyles</code> | 31 |
| <code>\glsname</code> | 76, 77 | <code>\glsshortaccessdisplay</code> | 206 |
| <code>\glsnameaccessdisplay</code> | 204 | <code>\glsshortaccesskey</code> | 219 |
| <code>\glsnamefont</code> | 129 | <code>\glsshortkey</code> | 133 |
| <code>\glsnavhyperlink</code> | 157 | <code>\glsshortpluralaccessdisplay</code> | 206 |
| <code>\glsnavhypertarget</code> | 157 | <code>\glsshortpluralaccesskey</code> | 219 |
| <code>\glsnavigation</code> | 158 | <code>\glsshortpluralkey</code> | 133 |
| <code>\glsnextpages</code> | 124 | <code>\glsshorttok</code> | 136 |
| <code>\glsnonextpages</code> | 123 | <code>\glssortnumberfmt</code> | 7 |
| <code>\glsnoxindywarning</code> | 24 | <code>\glsstepentry</code> | 124 |
| <code>\glsnumberformat</code> | 21 | <code>\glsstepsubentry</code> | 124 |
| <code>\glsnumbersgroupname</code> | 17, 106, 127 | <code>\glssubentrycounterlabel</code> | 125 |
| <code>\glsopenbrace</code> | 106 | <code>\glssubentryitem</code> | 125 |
| <code>\glsorder</code> | 13 | <code>\GLSsymbol</code> | 82 |
| <code>\glspagelistwidth</code> | 162, 168, 173, 180 | <code>\Glsymbol</code> | 81 |
| <code>\glspar</code> | 19 | <code>\glsymbol</code> | 81, 82 |
| <code>\GLSpl</code> | 68 | <code>\glsymbolaccessdisplay</code> | 205 |
| <code>\Glspl</code> | 67, 155 | <code>\glsymbolnav</code> | 159 |
| <code>\glspl</code> | 50, 51, 66–68 | <code>\GLSsymbolplural</code> | 83 |
| <code>\GLSplural</code> | 74 | <code>\Glsymbolplural</code> | 83 |
| <code>\Glsplural</code> | 74 | <code>\glsymbolplural</code> | 82, 83 |
| <code>\glsplural</code> | 73, 74 | <code>\glsymbolpluralaccessdisplay</code> | 205 |
| <code>\glspluralaccessdisplay</code> | 204 | <code>\glsymbolsgroupname</code> | 17, 106, 127 |
| <code>\glspluralsuffix</code> | 17, 39 | <code>\glstarget</code> | 126 |
| <code>\glspostdescription</code> | 19 | <code>\GLStext</code> | 71 |
| <code>\glsquote</code> | 106 | <code>\Glstext</code> | 72 |
| <code>\glsrefentry</code> | 125 | <code>\glstext</code> | 71 |
| <code>\glsreset</code> | 49 | <code>\glstextaccessdisplay</code> | 204 |
| <code>\glsresetall</code> | 49 | <code>\glstextformat</code> | 50 |
| <code>\glsresetentrylist</code> | 123 | <code>\glstreeindent</code> | 187, 188 |
| <code>\glsresetsubentrycounter</code> | 124 | <code>\glsunset</code> | 49 |
| <code>\glssee</code> | 118 | <code>\glsunsetall</code> | 49 |
| <code>\glsseeformat</code> | 109, 119 | <code>\GLSuseri</code> | 85 |
| <code>\glsseeitem</code> | 119 | <code>\Glsuseri</code> | 84 |
| <code>\glsseeitemformat</code> | 119 | <code>\glsuseri</code> | 84, 85 |
| <code>\glsseeleastsep</code> | 119 | <code>\GLSuserii</code> | 86 |
| <code>\glsseeelist</code> | 119 | <code>\Glsuserii</code> | 86 |
| <code>\glsseeesep</code> | 119 | <code>\glsuserii</code> | 85, 86 |
| <code>\glsSetAlphaCompositor</code> | 20 | <code>\GLSuseriii</code> | 88 |
| <code>\glsSetCompositor</code> | 20 | <code>\Glsuseriii</code> | 87 |
| <code>\glsSetSuffixF</code> | 21 | <code>\glsuseriii</code> | 87, 88 |
| <code>\glsSetSuffixFF</code> | 21 | <code>\GLSuseriv</code> | 89 |
| <code>\glssettoctitle</code> | 16 | <code>\Glsuseriv</code> | 89 |
| <code>\glssetwidest</code> | 189 | <code>\glsuseriv</code> | 88, 89 |
| <code>\GlsSetXdyCodePage</code> | 33 | <code>\GLSuserv</code> | 91 |

| | | | |
|--|------------------------------------|--|---|
| <code>\Glsuserv</code> | 90 | <code>long3colborder</code> (style) | 164 |
| <code>\glsuserv</code> | 90, 91 | <code>long3colheader</code> (style) | 165 |
| <code>\GLSuservi</code> | 92 | <code>long3colheaderborder</code> (style) | 165 |
| <code>\Glsuservi</code> | 92 | <code>long4col</code> (style) | 165 |
| <code>\glsuservi</code> | 91, 92 | <code>long4colborder</code> (style) | 166 |
| <code>\glswrite</code> | 115 | <code>long4colheader</code> (style) | 166 |
| <code>\glswritefiles</code> | 115 | <code>long4colheaderborder</code> (style) | 166 |
| H | | | |
| <code>\hyperbf</code> | 131 | <code>longaccess</code> (key) | 201 |
| <code>\hyperemph</code> | 131 | <code>longborder</code> (style) | 163 |
| <code>hyperfirst</code> (option) | 13 | <code>longheader</code> (style) | 163 |
| <code>\hyperit</code> | 131 | <code>longheaderborder</code> (style) | 163 |
| <code>\hyperlink</code> | 62 | <code>longplural</code> (key) | 41 |
| <code>\hypermd</code> | 131 | <code>longpluralaccess</code> (key) | 201 |
| <code>\hyperpage</code> | 129 | <code>longragged</code> (style) | 168 |
| <code>hyperref</code> package | 117, 120, 129, 155 | <code>longragged3col</code> (style) | 169 |
| <code>\hyperrm</code> | 131 | <code>longragged3colborder</code> (style) | 170 |
| <code>\hypersc</code> | 131 | <code>longragged3colheader</code> (style) | 170 |
| <code>\hypersf</code> | 131 | <code>longragged3colheaderborder</code> (style) | 171 |
| <code>\hypersl</code> | 131 | <code>longraggedborder</code> (style) | 169 |
| <code>\hypertarget</code> | 62 | <code>longraggedheader</code> (style) | 169 |
| <code>\hypertt</code> | 131 | <code>longraggedheaderborder</code> (style) | 169 |
| <code>\hyperup</code> | 131 | <code>longtable</code> (environment) 5, 150, 162–172 | |
| | | <code>longtable</code> package | 162, 168 |
| I | | | |
| <code>\if@glsisacronymlist</code> | 10 | M | |
| <code>\ifglossaryexists</code> | 35 | <code>\makefirststuc</code> | 155 |
| <code>\ifglsentryexists</code> | 35 | <code>makeglossaries</code> | |
| <code>\ifglsused</code> | 35, 48 | .. 13, 20, 31, 33, 36, 114, 120, 121 | |
| <code>\ifglsxindy</code> | 14 | <code>\makeglossaries</code> | 16, 19–21, 36, 37, 114 |
| <code>index</code> (style) | 185 | <code>\makeglossary</code> | 114 |
| <code>indexgroup</code> (style) | 186 | <code>makeindex</code> | 12, 14, 17, |
| <code>indexhypergroup</code> (style) | 186 | .. 19–21, 36–39, 48, 55, 57, 106, | |
| <code>\inputencodingname</code> | 14 | .. 109, 111, 113, 117, 126, 127, 194 | |
| <code>\istfile</code> | 115 | <code>delim_n</code> | 21 |
| <code>\istfilename</code> | 20 | <code>delim_r</code> | 21 |
| <code>\item</code> | 129, 159, 185 | <code>page_compositor</code> | 20 |
| L | | | |
| <code>\languagename</code> | 31 | special characters | 56, 106 |
| <code>link text</code> | 50 | <code>memoir class</code> | 116 |
| <code>list</code> (style) | 159 | <code>mfirstuc</code> package | 1 |
| <code>listdotted</code> (style) | 161 | N | |
| <code>listgroup</code> (style) | 160 | <code>name</code> (key) | 38 |
| <code>listhypergroup</code> (style) | 160 | <code>\newacronym</code> | 13, 41, 50, 131, 132 |
| <code>\loadglsentries</code> | 9, 50 | <code>\newacronymhook</code> | 136 |
| <code>long</code> (key) | 41 | <code>\newglossary</code> | 11, 36, 37, 113, 114, 122 |
| <code>long</code> (style) | 162 | <code>\newglossaryentry</code> | 11, 42, 50, 132 |
| <code>long3col</code> (style) | 164 | <code>\newglossaryentry options</code> | |
| | | <code>access</code> | 201, 202 |
| | | <code>counter</code> | 40 |

| | | | |
|-------------------------|---|--------------------------|-------------------------|
| description | 11–13, 38, 42, 50, 51, 78, 100, 132, 142, 200 | nostyles (option) | 6 |
| descriptionaccess | 203, 205 | nosuper (option) | 6 |
| descriptionplural | 79, 200 | notree (option) | 6 |
| descriptionpluralaccess | 203, 206 | numberedsection (option) | 4 |
| first | 39, 44, 50, 63, 72, 102, 141, 144, 200 | numberline (option) | 4 |
| firstaccess | 202, 205 | | |
| firstplural | 39, 44, 50, 75, 102, 200 | O | |
| firstpluralaccess | 203, 205 | \oldacronym | 131 |
| format | 108 | order (option) | 14 |
| long | 104, 201 | | |
| longaccess | 203, 206 | P | |
| longplural | 104, 201 | package options: | |
| longpluralaccess | 204, 206 | acronym | 9, 16, 122, 131, 132 |
| name | 11, 12, 38, 39, 42, 76, 100, 119, 199 | acronym | 9 |
| nonumberlist | 40 | acronymlists | 11 |
| parent | 40, 42 | compatible-2.07 | 15 |
| plural | 39, 44, 51, 73, 200 | counter | 11 |
| pluralaccess | 203, 204 | counter | 11 |
| see | 5, 40 | description | 141, 142 |
| short | 104, 200 | description | 13 |
| shortaccess | 203, 206 | dua | 140–142 |
| shortplural | 104, 200 | dua | 13 |
| shortpluralaccess | 203, 206 | entrycounter | 124 |
| sort | 11, 12, 39, 102, 126, 127 | true | 6 |
| symbol | 11, 12, 38, 39, 50, 51, 81, 101, 138–141, 144, 165, 176, 199–201 | entrycounter | 6 |
| symbolaccess | 203, 205 | entrycounterwithin | 6 |
| symbolplural | 82, 200 | footnote | 64–70, 139–142, 207–212 |
| symbolpluralaccess | 203, 205 | footnote | 13 |
| text | 39, 51, 63, 71, 101, 138, 141, 200 | hyperfirst | |
| textaccess | 202, 204 | false | 64–70 |
| type | 9, 40, 50, 102 | hyperfirst | 13 |
| user1 | 84, 102, 201 | makeindex | 109, 155 |
| user2 | 85, 103 | nolist | 150 |
| user3 | 87, 103 | nolist | 6 |
| user4 | 88, 103 | nolong | 150, 162 |
| user5 | 90, 103 | nolong | 5 |
| user6 | 91, 103, 201 | nomain | 9 |
| \newglossarystyle | 128 | nonumberlist | 5 |
| ngerman package | 32 | nonumberlist | 5 |
| \noist | 112, 155, 199 | nostyles | 6 |
| nolist (option) | 6 | nosuper | 150 |
| nolong (option) | 5 | nosuper | 6 |
| nonumberlist (key) | 40 | notree | 150 |
| nonumberlist (option) | 5 | notree | 6 |
| \nopostdesc | 19 | numberedsection | 4 |
| | | numberline | 4 |
| | | numberline | 4 |
| | | order | 14 |
| | | sanitize | 11, 12, 38, 100, 101 |
| | | sanitize | 12 |

X

| | | | |
|---------------------------------|--|-----------------------------------|------------------------|
| <code>\xglsaccsupp</code> | 204 | 109, 117, 120, 121, 126, 155, 194 | |
| <code>xindy</code> | 12, 14, 19, 20, 24, 27, 29, 31, 33, 34, 47, 60, 61, 106, 107, | <code>\xmakefirstuc</code> | 157 |
| | | <code>xspace package</code> | 3, 132 |