

# Documented Code For glossaries v3.06

Nicola L.C. Talbot

Dickimaw Books

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

2013-06-17

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 v3.06:  $\TeX$ 2e Package to Assist Generating Glossaries”.

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

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

**INSTALL** Installation instructions.

**CHANGES** Change log.

**README** Package summary.

# Contents

<b>1</b>	<b>Main Package Code</b>	<b>3</b>
1.1	Package Definition	3
1.2	Package Options	5
1.3	Default values	20
1.4	Xindy	28
1.5	Loops and conditionals	37
1.6	Defining new glossaries	40
1.7	Defining new entries	42
1.8	Resetting and unsetting entry flags	54
1.9	Loading files containing glossary entries	55
1.10	Using glossary entries in the text	56
1.10.1	Links to glossary entries	57
1.10.2	Displaying entry details without adding information to the glossary	109
1.11	Adding an entry to the glossary without generating text	115
1.12	Creating associated files	116
1.13	Writing information to associated files	125
1.14	Glossary Entry Cross-References	130
1.15	Displaying the glossary	132
1.16	Acronyms	145
1.17	Predefined acronym styles	149
1.18	Predefined Glossary Styles	164
1.19	Debugging Commands	165
1.20	Compatibility with version 2.07 and below	169
<b>2</b>	<b>Mfirstuc Documented Code</b>	<b>169</b>
<b>3</b>	<b>Glossary Styles</b>	<b>172</b>
3.1	Glossary hyper-navigation definitions (glossary-hypernav package)	172
3.2	In-line Style (glossary-inline.sty)	174
3.3	List Style (glossary-list.sty)	176
3.4	Glossary Styles using longtable (the glossary-long package)	179
3.5	Glossary Styles using longtable (the glossary-longragged package)	185
3.6	Glossary Styles using multicol (glossary-mcols.sty)	190
3.7	Glossary Styles using supertabular environment (glossary-super package)	194
3.8	Glossary Styles using supertabular environment (glossary-superragged package)	200
3.9	Tree Styles (glossary-tree.sty)	206
<b>4</b>	<b>glossaries-compatible-207</b>	<b>214</b>

<b>5 Accessibility Support (glossaries-accsupp Code)</b>	<b>220</b>
5.1 Defining Replacement Text . . . . .	221
5.2 Accessing Replacement Text . . . . .	224
5.3 Displaying the Glossary . . . . .	236
5.4 Acronyms . . . . .	237
5.5 Debugging Commands . . . . .	240
<b>6 Multi-Lingual Support</b>	<b>242</b>
6.1 Babel Captions . . . . .	242
6.2 Polyglossia Captions . . . . .	248
6.3 Brazilian Dictionary . . . . .	251
6.4 Danish Dictionary . . . . .	251
6.5 Dutch Dictionary . . . . .	252
6.6 English Dictionary . . . . .	252
6.7 French Dictionary . . . . .	252
6.8 German Dictionary . . . . .	252
6.9 Irish Dictionary . . . . .	253
6.10 Italian Dictionary . . . . .	253
6.11 Magyar Dictionary . . . . .	253
6.12 Polish Dictionary . . . . .	254
6.13 Serbian Dictionary . . . . .	254
6.14 Spanish Dictionary . . . . .	254
<b>Glossary</b>	<b>254</b>
<b>Change History</b>	<b>255</b>
<b>Index</b>	<b>265</b>

# 1 Main Package Code

## 1.1 Package Definition

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

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries}[2013/06/17 v3.06 (NLCT)]
```

Required packages:

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

7 \RequirePackage{datatool-base}
```

Need to use `\new@ifnextchar` instead of `\@ifnextchar` in commands that have a final optional argument (such as `\gls`) so require . Thanks to Morten Høgholm for suggesting this. (This has replaced using the `xspace` package.)

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

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

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

Check if doc has been loaded.

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

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

18 \if@gls@docloaded
```

It has been loaded, so some modifications need to be made to ensure both packages can work together.

```
\glsorg@glossary First, save the original behaviour of \glossary
```

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

```
\glsorg@wrglossary
```

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

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

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

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

`\glsorg@theglossary`

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

`sorg@endtheglossary`

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

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

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

End of doc stuff.

```
47 \fi
```

## 1.2 Package Options

`toc` The `toc` package option will add the glossaries to the table of contents. This is a boolean key, if the value is omitted it is taken to be true.

```
48 \define@boolkey{glossaries.sty}[gls]{toc}[true]{}
```

`numberline` The `numberline` package option adds `\numberline` to `\addcontentsline`. Note that this option only has an effect if used in with `toc=true`.

```
49 \define@boolkey{glossaries.sty}[gls]{numberline}[true]{}
```

`\@@glossarysec` The sectional unit used to start the glossary is stored in `\@@glossarysec`. If chapters are defined, this is initialised to `chapter`, otherwise it is initialised to `section`.

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

`section` The `section` key can be used to set the sectional unit. If no unit is specified, use `section` as the default. The starred form of the named sectional unit will be used. If you want some other way to start the glossary section (e.g. a numbered section) you will have to redefined `\glossarysection`.

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

Determine whether or not to use numbered sections.

```

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

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

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

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

```

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

```

ssary@default@style
73 \newcommand*{\@glossary@default@style}{list}

style The default glossary style can be changed using the style package option. The
value can be the name of any defined glossary style. The glossary style is set at
the beginning of the document, so you can still use the style key to set a style
that is defined in another package. This package comes with some predefined
styles that are defined in subsection 1.18.
74 \define@key{glossaries.sty}{style}{%
75 \renewcommand*{\@glossary@default@style}{#1}}

```

Each entry within a given glossary will have an associated number list. By default, this refers to the page numbers on which that entry has been used, but it can also refer to any counter used in the document (such as the section or equation counters). The default number list format displays the number list “as is”:

```

lossaryentrynumbers
76 \newcommand*{\glossaryentrynumbers}[1]{#1\gls@save@numberlist{#1}}

```

`nonumberlist` Note that the entire number list for a given entry will be passed to `\glossaryentrynumbers` so any font changes will also be applied to the delimiters. The `nonumberlist` package option suppresses the number lists (this simply redefines `\glossaryentrynumbers` to ignore its argument).

```

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

```

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

```

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

```

`o@seeautonumberlist`

```

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

```

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

```

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

```

`\@gls@loadlong`

```

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

```

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

```

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

```

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

```

90 \IfFileExists{supertabular.sty}{%
91   \newcommand*{\@gls@loadsuper}{\RequirePackage{glossary-super}}}%
92   \newcommand*{\@gls@loadsuper}{} }

```

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

```

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

```

`\@gls@loadlist`

```

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

```

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

```

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

```

`\@gls@loadtree`

```

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

notree This option prevents from being loaded (to reduce overheads if required). Nat-
        97 \DeclareOptionX{notree}{\renewcommand*{\@gls@loadtree}{} }
        98 \DeclareOptionX{nostyles}{%
        99   \renewcommand*{\@gls@loadlong}{}%
       100   \renewcommand*{\@gls@loadsuper}{}%
       101   \renewcommand*{\@gls@loadlist}{}%
       102   \renewcommand*{\@gls@loadtree}{}%
       103   \let\@glossary@default@style\relax
       104 }

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

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

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

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

entrycounter Defines a counter that can be used in the standard glossary styles to number
            each (main) entry. If true, this will define a counter called glossaryentry.
       112 \define@boolkey{glossaries.sty}[gls]{entrycounter}[true]{}
       113 \glsentrycounterfalse

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

```

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

```
118 \newcommand*\@gls@counterwithin{}\}
```

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

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

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

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

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

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

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

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

Set standard sort as the default:

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

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

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

```

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

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

```

Set index information for this entry

```

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

```

`\glsdefmain` Define the main glossary. This will be the first glossary to be displayed when using `\printglossaries`. The default extensions conflict if used with `doc`, so provide different extensions if `doc` loaded. (If these extensions are inappropriate, use `nomain` and manually define the main glossary with the desired extensions.)

```

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

```

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

`\glsdefaulttype`

```

194 \newcommand*{\glsdefaulttype}{main}

```

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

`\acronymtype`

```

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

```

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

```

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

```

acronym The acronym option sets an associated conditional which is used in [subsection 1.16](#) to determine whether or not to define a separate glossary for acronyms.

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

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

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

\@addtoacronymlists

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

\DeclareAcronymList Identifies the named glossary as a list of acronyms and adds to the list. (Doesn't check if the glossary exists, but checks if label already in list. Use \SetAcronymStyle after identifying all the acronym lists.)

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

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

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

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

Internal command requires label and list to be expanded:

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

Not found

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

Found

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

`\if@glsisacronymlist` Convenient boolean.

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

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

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

`\SetAcronymLists` Sets the “list of acronyms” list. Argument must be a comma-separated list of glossary labels. (Doesn’t check at this point if the glossaries exists.)

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

`acronymlists`

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

The default counter associated with the numbers in the glossary is stored in `\glscounter`. This is initialised to the page counter. This is used as the default counter when a new glossary is defined, unless a different counter is specified in the optional argument to `\newglossary` (see [subsection 1.6](#)).

`\glscounter`

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

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

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

`\@gls@nohyperlist`

```
244 \newcommand*\@gls@nohyperlist{}
```

`\DeclareNoHyperList`

```
245 \newcommand*\GlsDeclareNoHyperList[1]{%
246   \ifdefempty\@gls@nohyperlist
247   {%
248     \renewcommand*\@gls@nohyperlist{#1}%
249   }%
250   {%
```

```

251 \appto\@gls@nohyperlist{, #1}%
252 }%
253 }

```

nohypertypes

```

254 \define@key{glossaries.sty}{nohypertypes}{%
255 \GlsDeclareNoHyperList{#1}%
256 }

```

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

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

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

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

```

Before defining the `sanitize` package option, The key-value list for the `sanitize` value needs to be defined. These are all boolean keys. If they are not given a value, assume true.

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

```

261 \define@boolkey[glS]{sanitize}{description}[true]{%
262 \ifglS@sanitize@description
263   \renewcommand*{\@glS@sanitizedesc}{\@onelevel@sanitize\@glo@desc}%
264 \else
265   \renewcommand*{\@glS@sanitizedesc}{}%
266 \fi
267 }

```

Similarly for the name key:

```

268 \define@boolkey[glS]{sanitize}{name}[true]{%
269 \ifglS@sanitize@name
270   \renewcommand*{\@glS@sanitizename}{\@onelevel@sanitize\@glo@name}%
271 \else
272   \renewcommand*{\@glS@sanitizename}{}%
273 \fi}

```

and for the symbol key:

```

274 \define@boolkey[glS]{sanitize}{symbol}[true]{%
275 \ifglS@sanitize@symbol
276   \renewcommand*{\@glS@sanitizesymbol}{%
277 \@onelevel@sanitize\@glo@symbol}%
278 \else
279   \renewcommand*{\@glS@sanitizesymbol}{}%
280 \fi}

```

and for the sort key:

```

281 \define@boolkey[glS]{sanitize}{sort}[true]{%
282 \ifglS@sanitize@sort
283   \renewcommand*{\@glS@sanitizesort}{%
284 \@onelevel@sanitize\@glo@sort}%
285 \else
286   \renewcommand*{\@glS@sanitizesort}{}%
287 \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:

```

288 \define@key{glossaries.sty}{sanitize}[description=true,symbol=true,
289 name=true]{%
290 \ifthenelse{\equal{#1}{none}}{%
291 {%
292   \renewcommand*{\@glS@sanitizedesc}{}%
293   \renewcommand*{\@glS@sanitizename}{}%
294   \renewcommand*{\@glS@sanitizesymbol}{}%
295 }%
296 {%
297   \setkeys[glS]{sanitize}{#1}}%
298 }

```

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

```

299 \define@boolkey{glossaries.sty}[glS]{translate}[true]{%

```

Set the default value:

```
300 \glstranslatefalse
301 \ifpackageloaded{translator}%
302   {\glstranslatetrue}%
303   {%
304     \ifpackageloaded{polyglossia}%
305       {\glstranslatetrue}%
306       {%
307         \ifpackageloaded{babel}{\glstranslatetrue}{}}%
308       }%
309 }
```

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

```
310 \define@boolkey{glossaries.sty}[gls]{indexonlyfirst}[true]{}
311 \glindexonlyfirstfalse
```

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

```
312 \define@boolkey{glossaries.sty}[gls]{hyperfirst}[true]{}
313 \glshyperfirsttrue
```

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

```
314 \define@boolkey{glossaries.sty}[glsacr]{footnote}[true]{}%
315 \ifthenelse{\boolean{glsacrdescription}}{}{}%
316 {\renewcommand*{\@gls@sanitizedesc}{}}%
317 }
```

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

```
318 \define@boolkey{glossaries.sty}[glsacr]{description}[true]{}%
319 \renewcommand*{\@gls@sanitizesymbol}{}%
320 }
```

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

```
321 \define@boolkey{glossaries.sty}[glsacr]{smallcaps}[true]{}%
322 \renewcommand*{\@gls@sanitizesymbol}{}%
323 }
```

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

```
324 \define@boolkey{glossaries.sty}[glsacr]{smaller}[true]{}%
325 \renewcommand*{\@gls@sanitizesymbol}{}%
326 }
```

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

```
327 \define@boolkey{glossaries.sty}[glsacr]{dua}[true]{}%
328 \renewcommand*{\@gls@sanitizesymbol}{}%
329 }
```

`shortcuts` Define acronym shortcuts.  
330 `\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.  
331 `\newcommand*{\glsorder}{word}`

`\@glsorder` The ordering information is written to the auxiliary file for `makeglossaries`, so ignore the auxiliary information.  
332 `\newcommand*{\@glsorder}[1]{}`

`order`  
333 `\define@choicekey{glossaries.sty}{order}{word,letter}{%`  
334 `\def\glsorder{#1}}`

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

The default is `makeindex`:  
336 `\glxindyfalse`

Define package option to specify that `makeindex` will be used to sort the glossaries:  
337 `\DeclareOptionX{makeindex}{\glxindyfalse}`

The `xindy` package option may have a value which in turn can be a key=value list. First define the keys for this sub-list. The boolean `glsnumbers` determines whether to automatically add the `glsnumbers` letter group.  
338 `\define@boolkey[glx]{xindy}{glsnumbers}[true]{}`  
339 `\glx@xindy@glsnumberstrue`

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

Define key to set the language  
341 `\define@key[glx]{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.  
342 `\ifcsundef{inputencodingname}{%`  
343 `\def\gls@codepage{}}{%`  
344 `\def\gls@codepage{\inputencodingname}`  
345 `}`

Define a key to set the code page.

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

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

```
347 \define@key{glossaries.sty}{xindy}[]{%
348   \glSxindytrue
349   \setkeys[glS]{xindy}{#1}%
350 }
```

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

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

Set default:

```
352 \glSsavewritesfalse
```

**\GlossariesWarning** Prints a warning message.

```
353 \newcommand*{\GlossariesWarning}[1]{%
354   \PackageWarning{glossaries}{#1}%
355 }
```

**sariesWarningNoLine** Prints a warning message without the line number.

```
356 \newcommand*{\GlossariesWarningNoLine}[1]{%
357   \PackageWarningNoLine{glossaries}{#1}%
358 }
```

Define package option to suppress warnings

```
359 \DeclareOptionX{nowarn}{%
360   \renewcommand*{\GlossariesWarning}[1]{}%
361   \renewcommand*{\GlossariesWarningNoLine}[1]{}%
362 }
```

**compatible-2.07**

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

Process package options:

```
365 \ProcessOptionsX
```

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

```
366 \ifglStranslate
367   \@ifpackageloaded{polyglossia}%
368   {%
```

polyglossia fakes babel so need to check for polyglossia first.

```

369 }%
370 {%
371   \@ifpackageloaded{babel}%
372   {%
373     \IfFileExists{translator.sty}%
374     {%
375       \RequirePackage{translator}%
376     }%
377   }%
378 }%
379 {}
380 }
381 \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}).

```

382 \ifthenelse{\equal{\glscounter}{section}}{%
383 {%
384   \ifcsundef{chapter}{}%
385   {%
386     \let\@gls@old@chapter\@chapter
387     \def\@chapter[#1]#2{\@gls@old@chapter[#1]{#2}%
388       \ifcsundef{hyperdef}{\hyperdef{section}{\thesection}}}%
389   }%
390 }%
391 {}

```

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

```

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

```

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

```

393 \newcommand*{\@onlypremakeg}[1]{%
394 \ifx\@gls@onlypremakeg\@empty
395   \def\@gls@onlypremakeg{#1}%
396 \else
397   \expandafter\toks@\expandafter{\@gls@onlypremakeg}%
398   \edef\@gls@onlypremakeg{\the\toks@,\noexpand#1}%
399 \fi}

```

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

```
400 \newcommand*{\@disable@onlypremakeg}{%
401 \@for\@thiscs:=\@gls@onlypremakeg\do{%
402   \expandafter\@disable@premakecs\@thiscs%
403 }}
```

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

```
404 \newcommand*{\@disable@premakecs}[1]{%
405   \def#1{\PackageError{glossaries}{\string#1\space may only be
406     used before \string\makeglossaries}{You can't use
407     \string#1\space after \string\makeglossaries}}%
408 }
```

### 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`

```
409 \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`

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

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

```
411 \newcommand*{\glssettoctitle}[1]{%
412 \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`

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

`\descriptionname`

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

`\symbolname`

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

`\pagelistname`

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

Labels for makeindex's symbol and number groups:

glsymbolsgroupname

```
417 \providecommand*{\glsymbolsgroupname}{Symbols}
```

glsnumbersgroupname

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

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

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

\seename

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

\andname

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

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

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

```
422 \newcommand*{\addglossarytocaptions}[1]{%
423   \ifcsundef{captions#1}{}%
424   {%
425     \expandafter\let\expandafter\@gls@tmp\csname captions#1\endcsname
426     \expandafter\toks@\expandafter{\@gls@tmp
427       \renewcommand*{\glossaryname}{\translate{Glossary}}}%
428     }%
429     \expandafter\edef\csname captions#1\endcsname{\the\toks@}%
430   }%
431 }
```

```
432 \ifglstranslate
```

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

```
433 \@ifpackageloaded{translator}{%
434   \usedictionary{glossaries-dictionary}%
435   \addglossarytocaptions{portuges}%
436   \addglossarytocaptions{portuguese}%
437   \addglossarytocaptions{brazil}%
438   \addglossarytocaptions{brazilian}%
439   \addglossarytocaptions{danish}%
440   \addglossarytocaptions{dutch}%
441   \addglossarytocaptions{afrikaans}%
442   \addglossarytocaptions{english}%
443   \addglossarytocaptions{UKenglish}%

```

```

444 \addglossarytocaptions{USenglish}%
445 \addglossarytocaptions{american}%
446 \addglossarytocaptions{australian}%
447 \addglossarytocaptions{british}%
448 \addglossarytocaptions{canadian}%
449 \addglossarytocaptions{newzealand}%
450 \addglossarytocaptions{french}%
451 \addglossarytocaptions{frenchb}%
452 \addglossarytocaptions{français}%
453 \addglossarytocaptions{acadian}%
454 \addglossarytocaptions{canadien}%
455 \addglossarytocaptions{german}%
456 \addglossarytocaptions{germanb}%
457 \addglossarytocaptions{austrian}%
458 \addglossarytocaptions{naustrian}%
459 \addglossarytocaptions{ngerman}%
460 \addglossarytocaptions{irish}%
461 \addglossarytocaptions{italian}%
462 \addglossarytocaptions{magyar}%
463 \addglossarytocaptions{hungarian}%
464 \addglossarytocaptions{polish}%
465 \addglossarytocaptions{spanish}%
466 \renewcommand*{\glstoc{title}}[1]{%
467 \ifthenelse{\equal{#1}{main}}{%
468 \translatelet{\glossarytoc{title}}{Glossary}}{%
469 \ifthenelse{\equal{#1}{acronym}}{%
470 \translatelet{\glossarytoc{title}}{Acronyms}}{%
471 \def\glossarytoc{title}{\csname @glotype@#1@title\endcsname}}}%
472 \renewcommand*{\glossaryname}{\translate{Glossary}}%
473 \renewcommand*{\acronymname}{\translate{Acronyms}}%
474 \renewcommand*{\entryname}{\translate{Notation (glossaries)}}%
475 \renewcommand*{\descriptionname}{%
476 \translate{Description (glossaries)}}%
477 \renewcommand*{\symbolname}{\translate{Symbol (glossaries)}}%
478 \renewcommand*{\pagelistname}{%
479 \translate{Page List (glossaries)}}%
480 \renewcommand*{\glssymbolsgroupname}{%
481 \translate{Symbols (glossaries)}}%
482 \renewcommand*{\glnumbersgroupname}{%
483 \translate{Numbers (glossaries)}}%
484 }{%

485 \@ifpackageloaded{polyglossia}%
486 {\RequirePackage{glossaries-polyglossia}}%
487 {%
488 \@ifpackageloaded{babel}{%
489 \RequirePackage{glossaries-babel}}}%
490 }}
491 \fi

```

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

```
492 \DeclareRobustCommand*\nopostdesc{}\}
```

`\@nopostdesc` Suppress next description terminator.

```
493 \newcommand*\@nopostdesc{%
494   \let\org@glspostdescription\glspostdescription
495   \def\glspostdescription{%
496     \let\glspostdescription\org@glspostdescription}%
497 }
```

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

```
498 \newcommand{\glspar}{\par}
```

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

```
499 \ifglsxindy
500   \newcommand{\setStyleFile}[1]{%
501     \renewcommand{\istfilename}{#1.xdy}}
502 \else
503   \newcommand{\setStyleFile}[1]{%
504     \renewcommand{\istfilename}{#1.ist}}
505 \fi
```

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

```
506 \@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`

```
507 \ifglsxindy
508   \def\istfilename{\jobname.xdy}
509 \else
510   \def\istfilename{\jobname.ist}
511 \fi
```

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

`\@istfilename`

```
512 \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`

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

`\glsSetCompositor` Sets the compositor.

```
514 \newcommand*\glsSetCompositor[1]{%
515   \renewcommand*\glscompositor{#1}}
Only use before \makeglossaries
516 \@onlypremakeg\glsSetCompositor
```

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

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

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

`glsSetAlphaCompositor` Sets the alpha compositor.

```
518 \ifglsxindy
519   \newcommand*\glsSetAlphaCompositor[1]{%
520     \renewcommand*\@glsAlphacompositor{#1}}
521 \else
522   \newcommand*\glsSetAlphaCompositor[1]{%
523     \glsnoxindywarning\glsSetAlphaCompositor}
524 \fi
Can only be used before \makeglossaries
525 \@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.

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

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

```
527 \newcommand*\glsSetSuffixF[1]{%
528   \renewcommand*\gls@suffixF{#1}}
Only has an effect when used before \makeglossaries
529 \@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.

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

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

```
531 \newcommand*{\glsSetSuffixFF}[1]{%  
532   \renewcommand*{\gls@suffixFF}{#1}%  
533 }
```

`\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".

```
534 \ifcsundef{hyperlink}%  
535 {%  
536   \newcommand*{\glsnumberformat}[1]{#1}%  
537 }%  
538 {%  
539   \newcommand*{\glsnumberformat}[1]{\glshypernumber{#1}}%  
540 }
```

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`

```
541 \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`

```
542 \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`

```
543 \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`

```
544 \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`.

```
545 \newcommand*{\glossarysection}[2][\@gls@title]{%
546   \def\@gls@title{#2}%
547   \ifcsundef{phantomsection}%
548     {%
549       \@glossarysection{#1}{#2}%
550     }%
551   {%
552     \p@glossarysection{#1}{#2}%
553   }%
554   \glossarymark{\glossarytoctitle}%
555 }
```

`\glossarymark`

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

```
556 \ifcsundef{glossarymark}%
557 {%
558   \ifglsucmark
559     \newcommand{\glossarymark}[1]{%
560       \@mkboth{\MakeUppercase{#1}}{\MakeUppercase{#1}}%
561     }
562   \else
563     \newcommand{\glossarymark}[1]{\@mkboth{#1}{#1}}
564   \fi
565 }%
566 {%
567   \GlossariesWarning{overriding \string\glossarymark}%
568   \@ifclassloaded{memoir}%
569   {
570     \ifglsucmark
571       \renewcommand{\glossarymark}[1]{%
572         \@mkboth{\MakeUppercase{#1}}{\MakeUppercase{#1}}%
573       }
574     \else
```

```

575     \renewcommand{\glossarymark}[1]{%
576         \markboth{\memUHead{#1}}{\memUHead{#1}}%
577     }
578     \fi
579 }
580 {
581     \ifglsucmark
582         \renewcommand{\glossarymark}[1]{%
583             \@mkboth{\MakeUppercase{#1}}{\MakeUppercase{#1}}%
584         }
585     \else
586         \renewcommand{\glossarymark}[1]{\@mkboth{#1}{#1}}
587     \fi
588 }
589 }

```

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`

```

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

```

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

`\@glossarysection`

```

592 \newcommand*\@glossarysection[2]{%
593 \ifx\@glossarysecstar\@empty
594     \csname\@glossarysec\endcsname{#2}%
595 \else
596     \csname\@glossarysec\endcsname*{#2}%
597     \@gls@toc{#1}{\@glossarysec}%
598 \fi
599 \@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`

```

600 \newcommand*\@pglossarysection[2]{%
601 \glsclearpage
602 \phantomsection
603 \ifx\@glossarysecstar\@empty

```

```

604 \csname\@@glossarysec\endcsname{#2}%
605 \else
606 \@gls@toc{#1}{\@@glossarysec}%
607 \csname\@@glossarysec\endcsname*{#2}%
608 \fi
609 \@@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.

```

610 \newcommand*\gls@doclearpage{%
611 \ifthenelse{\equal{\@@glossarysec}{chapter}}%
612 {%
613 \ifcsundef{cleardoublepage}{\clearpage}{\cleardoublepage}%
614 }%
615 {}%
616 }

```

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

```

617 \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`

```

618 \newcommand*\@gls@toc[2]{%
619 \ifglstoc
620 \ifglsnumberline
621 \addcontentsline{toc}{#2}{\numberline{#1}}%
622 \else
623 \addcontentsline{toc}{#2}{#1}%
624 \fi
625 \fi}

```

## 1.4 Xindy

This section defines commands that only have an effect if `xindy` is used to sort the glossaries.

`\glsnnoxindywarning` Issues a warning if `xindy` hasn't been specified. These warnings can be suppressed by redefining `\glsnnoxindywarning` to ignore its argument

```

626 \newcommand*\glsnnoxindywarning[1]{%
627 \GlossariesWarning{Not in xindy mode --- ignoring \string#1}%
628 }

```

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

```
629 \ifglxindy
630   \edef\@xdyattributes{\string"default\string"}%
631 \fi
```

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

```
632 \ifglxindy
633   \edef\@xdyattributelist{}%
634 \fi
```

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

```
635 \ifglxindy
636   \def\@xdylocref{}
637 \fi
```

`\@gls@ifinlist`

```
638 \newcommand*\@gls@ifinlist[4]{%
639   \def\@do@ifinlist##1,#1,##2\end@do@ifinlist{%
640     \def\@gls@listsuffix{##2}%
641     \ifx\@gls@listsuffix\@empty
642       #4%
643     \else
644       #3%
645     \fi
646   }%
647   \@do@ifinlist,#2,#1,\end@do@ifinlist
648 }
```

`\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.

```
649 \ifglxindy
650   \newcommand*\@xdycounters{\@glscounter}
651   \newcommand*\GlsAddXdyCounters[1]{%
652     \@for\@gls@ctr:=#1\do{%
        Check if already in list before adding.
653       \edef\@do@addcounter{%
654         \noexpand\@gls@ifinlist{\@gls@ctr}{\@xdycounters}{}%
655         {%
656           \noexpand\edef\noexpand\@xdycounters{\@xdycounters,%
657             \noexpand\@gls@ctr}%
658         }%
659       }%
660       \@do@addcounter
661     }
662   }
```

Only has an effect before `\writeist`:

```
663 \onlypremakeg\GlsAddXdyCounters
664 \else
665 \newcommand*\GlsAddXdyCounters[1]{%
666 \glsnnoxindywarning\GlsAddXdyAttribute
667 }
668 \fi
```

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

```
669 \newcommand*\@disabled@glssaddxdycounters{%
670 \PackageError{glossaries}{\string\GlsAddXdyCounters\space
671 can't be used after \string\GlsAddXdyAttribute}{Move all
672 occurrences of \string\GlsAddXdyCounters\space before the first
673 instance of \string\GlsAddXdyAttribute}%
674 }
```

`\GlsAddXdyAttribute` Adds an attribute.

```
675 \ifglxindy
```

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

```
676 \newcommand*\@glssaddxdyattribute[2]{%
```

Add to xindy attribute list

```
677 \edef\@xdyattributes{\@xdyattributes ^^J \string"#1\string" ^^J
678 \string"#2#1\string"}%
```

Add to xindy markup location.

```
679 \expandafter\toks@\expandafter{\@xdylocref}%
680 \edef\@xdylocref{\the\toks@ ^^J}%
681 (markup-locref
682 :open \string"\string~n%
683 \expandafter\string\csname glsX#2X#1\endcsname
684 \string" ^^J
685 :close \string"\string" ^^J
686 :attr \string"#2#1\string"))%
```

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

```
687 \expandafter\gdef\csname glsX#2X#1\endcsname##1##2{%
688 \setentrycounter[##1]{#2}\csname #1\endcsname{##2}%
689 }%
690 }
```

High-level command:

```
691 \newcommand*\GlsAddXdyAttribute[1]{%
```

Add to comma-separated attribute list

```
692 \ifx\@xdyattributelist\@empty
693 \edef\@xdyattributelist{#1}%
694 \else
695 \edef\@xdyattributelist{\@xdyattributelist,#1}%
696 \fi
```

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

```

697 \for\@this@counter:=\@xdycounters\do{%
698 \protected@edef\gls@do@addxdyattribute{%
699 \noexpand\@glsaddxdyattribute{#1}{\@this@counter}%
700 }
701 \gls@do@addxdyattribute
702 }%

```

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

```

703 \let\GlsAddXdyCounters\@disabled@glsaddxdycounters
704 }

```

Only has an effect before `\writeist`:

```

705 \@onlypremakeg\GlsAddXdyAttribute
706 \else
707 \newcommand*\GlsAddXdyAttribute[1]{%
708 \glsnoxindywarning\GlsAddXdyAttribute}
709 \fi

```

redefinedattributes Add known attributes for all defined counters

```

710 \ifglxsindy
711 \newcommand*\@gls@addpredefinedattributes{%
712 \GlsAddXdyAttribute{glsnumberformat}
713 \GlsAddXdyAttribute{textrm}
714 \GlsAddXdyAttribute{textsf}
715 \GlsAddXdyAttribute{texttt}
716 \GlsAddXdyAttribute{textbf}
717 \GlsAddXdyAttribute{textmd}
718 \GlsAddXdyAttribute{textit}
719 \GlsAddXdyAttribute{textup}
720 \GlsAddXdyAttribute{textsl}
721 \GlsAddXdyAttribute{textsc}
722 \GlsAddXdyAttribute{emph}
723 \GlsAddXdyAttribute{glshypernumber}
724 \GlsAddXdyAttribute{hyperrrm}
725 \GlsAddXdyAttribute{hypersf}
726 \GlsAddXdyAttribute{hypertt}
727 \GlsAddXdyAttribute{hyperbf}
728 \GlsAddXdyAttribute{hypermd}
729 \GlsAddXdyAttribute{hyperit}
730 \GlsAddXdyAttribute{hyperup}
731 \GlsAddXdyAttribute{hypersl}
732 \GlsAddXdyAttribute{hypersc}
733 \GlsAddXdyAttribute{hyperemph}
734 }
735 \else
736 \let\@gls@addpredefinedattributes\relax
737 \fi

```

\@xdyuseralphabets List of additional alphabets

```
738 \def\@xdyuseralphabets{}
```

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

```
739 \ifglxindy
740   \newcommand*{\GlsAddXdyAlphabet}[2]{%
741     \edef\@xdyuseralphabets{%
742       \@xdyuseralphabets ^^J
743       (define-alphabet "#1" (#2))}}
744 \else
745   \newcommand*{\GlsAddXdyAlphabet}[2]{%
746     \glsnnoxindywarning\GlsAddXdyAlphabet}
747 \fi
```

This code is only required for xindy:

```
748 \ifglxindy
```

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

```
749   \newcommand*{\@gls@xdy@locationlist}{%
750     roman-page-numbers,%
751     Roman-page-numbers,%
752     arabic-page-numbers,%
753     alpha-page-numbers,%
754     Alpha-page-numbers,%
755     Appendix-page-numbers,%
756     arabic-section-numbers%
757   }
```

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

`\@roman-page-numbers` Lower case Roman numerals (i, ii, ...). In the event that `\roman` has been redefined to produce a fancy form of roman numerals, attempt to work out how it will be written to the output file.

```
758   \protected@edef\@gls@roman{\@roman{0}\string"
759     \string"roman-numbers-lowercase\string" :sep \string"}}%
760   \@onelevel@sanitize\@gls@roman
761   \edef\@tmp{\string" \string"roman-numbers-lowercase\string"
762     :sep \string"}%
763   \@onelevel@sanitize\@tmp
764   \ifx\@tmp\@gls@roman
765     \expandafter
766       \edef\csname @gls@xdy@Lclass@roman-page-numbers\endcsname{%
767         \string"roman-numbers-lowercase\string"%
768       }%
769   \else
770     \expandafter
771     \edef\csname @gls@xdy@Lclass@roman-page-numbers\endcsname{
```

```

772         :sep \string"\@gls@roman\string"%
773     }%
774 \fi

@Roman-page-numbers Upper case Roman numerals (I, II, ...).
775 \expandafter\def\csname @gls@xdy@Lclass@Roman-page-numbers\endcsname{%
776     \string"roman-numbers-uppercase\string"%
777 }%

arabic-page-numbers Arabic numbers (1, 2, ...).
778 \expandafter\def\csname @gls@xdy@Lclass@arabic-page-numbers\endcsname{%
779     \string"arabic-numbers\string"%
780 }%

@alpha-page-numbers Lower case alphabetical (a, b, ...).
781 \expandafter\def\csname @gls@xdy@Lclass@alpha-page-numbers\endcsname{%
782     \string"alpha\string"%
783 }%

@Alpha-page-numbers Upper case alphabetical (A, B, ...).
784 \expandafter\def\csname @gls@xdy@Lclass@Alpha-page-numbers\endcsname{%
785     \string"ALPHA\string"%
786 }%

appendix-page-numbers Appendix style locations (e.g. A-1, A-2, ..., B-1, B-2, ...). The separator is given
by \@glsAlphacompositor.
787 \expandafter\def\csname @gls@xdy@Lclass@Appendix-page-numbers\endcsname{%
788     \string"ALPHA\string"
789     :sep \string"\@glsAlphacompositor\string"
790     \string"arabic-numbers\string"%
791 }

arabic-section-numbers Section number style locations (e.g. 1.1, 1.2, ...). The compositor is given by
\glscompositor.
792 \expandafter\def\csname @gls@xdy@Lclass@arabic-section-numbers\endcsname{%
793     \string"arabic-numbers\string"
794     :sep \string"\glscompositor\string"
795     \string"arabic-numbers\string"%
796 }%

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

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

End of xindy-only block:
799 \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.)

```

800 \ifglxindy
801   \newcommand*{\GlsAddXdyLocation}[3][{}]{%
802     \def\@gls@tmp{#1}%
803     \ifx\@gls@tmp\@empty
804       \edef\@xdyuserlocationdefs{%
805         \@xdyuserlocationdefs ^^J%
806         (define-location-class \string"#2\string"^^J\space\space
807         \space(:sep \string"{}\glsoopenbrace\string" #3
808           :sep \string"\glsclosebrace\string"))
809       }%
810     \else
811       \edef\@xdyuserlocationdefs{%
812         \@xdyuserlocationdefs ^^J%
813         (define-location-class \string"#2\string"^^J\space\space
814         \space(:sep "\glsoopenbrace"
815           #1
816           :sep "\glsclosebrace\glsoopenbrace" #3
817           :sep "\glsclosebrace"))
818       }%
819     \fi
820     \edef\@xdyuserlocationnames{%
821       \@xdyuserlocationnames^^J\space\space\space
822       \string"#1\string"}%
823   }

```

Only has an effect before `\writeist`:

```

824   \@onlypremakeg\GlsAddXdyLocation
825 \else
826   \newcommand*{\GlsAddXdyLocation}[2]{%
827     \glsnnoxindywarning\GlsAddXdyLocation}
828 \fi

```

`\locationclassorder` Define location class order

```

829 \ifglxindy
830   \edef\@xdylocationclassorder{^^J\space\space\space
831     \string"roman-page-numbers\string"^^J\space\space\space
832     \string"arabic-page-numbers\string"^^J\space\space\space
833     \string"arabic-section-numbers\string"^^J\space\space\space
834     \string"alpha-page-numbers\string"^^J\space\space\space
835     \string"Roman-page-numbers\string"^^J\space\space\space
836     \string"Alpha-page-numbers\string"^^J\space\space\space
837     \string"Appendix-page-numbers\string"
838     \@xdyuserlocationnames^^J\space\space\space
839     \string"see\string"
840   }

```

841 \fi

Change the location order.

yLocationClassOrder

```
842 \ifglxindy
843   \newcommand*\GlsSetXdyLocationClassOrder[1]{%
844     \def\@xdylocationclassorder{#1}}
845 \else
846   \newcommand*\GlsSetXdyLocationClassOrder[1]{%
847     \glsnnoxindywarning\GlsSetXdyLocationClassOrder}
848 \fi
```

\@xdysortrules Define sort rules

```
849 \ifglxindy
850   \def\@xdysortrules{}
851 \fi
```

\GlsAddSortRule Add a sort rule

```
852 \ifglxindy
853   \newcommand*\GlsAddSortRule[2]{%
854     \expandafter\toks@\expandafter{\@xdysortrules}%
855     \protected@edef\@xdysortrules{\the\toks@ ^^J
856       (sort-rule \string"#1\string" \string"#2\string")}%
857   }
858 \else
859   \newcommand*\GlsAddSortRule[2]{%
860     \glsnnoxindywarning\GlsAddSortRule}
861 \fi
```

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

```
862 \ifglxindy
863   \def\@xdyrequiredstyles{tex}
864 \fi
```

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

```
865 \ifglxindy
866   \newcommand*\GlsAddXdyStyle[1]{%
867     \edef\@xdyrequiredstyles{\@xdyrequiredstyles,#1}}%
868 \else
869   \newcommand*\GlsAddXdyStyle[1]{%
870     \glsnnoxindywarning\GlsAddXdyStyle}
871 \fi
```

\GlsSetXdyStyles Reset the list of required styles

```
872 \ifglxindy
873   \newcommand*\GlsSetXdyStyles[1]{%
```

```

874 \edef\@xdyrequiredstyles{#1}}
875 \else
876 \newcommand*\GlsSetXdyStyles[1]{%
877 \glsnnoxindywarning\GlsSetXdyStyles}
878 \fi

```

`\findrootlanguage` This used to determine the root language, using a bit of trickery since babel doesn't supply the information, but now that babel is once again actively maintained, we can't do this any more, so `\findrootlanguage` no longer available. Now provide a command that does nothing (in case it's been patched).

```

879 \newcommand*\findrootlanguage{}{}

```

`\@xdylanguage` The xindy language setting is required by makeglossaries, so provide a command for makeglossaries to pick up the information from the auxiliary file. This command is not needed by the glossaries package, so define it to ignore its arguments.

```

880 \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.

```

881 \ifglsxindy
882 \newcommand*\GlsSetXdyLanguage[2][\glsdefaultttype]{%
883 \ifglossaryexists{#1}{%
884 \expandafter\def\csname @xdy@#1@language\endcsname{#2}%
885 }{%
886 \PackageError{glossaries}{Can't set language type for
887 glossary type '#1' --- no such glossary}{%
888 You have specified a glossary type that doesn't exist}}
889 \else
890 \newcommand*\GlsSetXdyLanguage[2][]{%
891 \glsnnoxindywarning\GlsSetXdyLanguage}
892 \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.

```

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

```

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

```

894 \ifglsxindy
895 \newcommand*\GlsSetXdyCodePage[1]{%
896 \renewcommand*\@gls@codepage{#1}%
897 }

```

Suggested by egreg:

```

898 \AtBeginDocument{
899   \ifx\gls@codepage\@empty
900   \@ifpackageloaded{fontspec}{\def\gls@codepage{utf8}}{}
901 \fi}
902 \else
903   \newcommand*\GlsSetXdyCodePage}[1]{%
904     \glsnoxindywarning\GlsSetXdyCodePage}
905 \fi

```

`\@xdylettergroups` Store letter group definitions.

```

906 \ifglxsxindy
907   \ifglsxindy@glsnumbers
908     \def\@xdylettergroups{(define-letter-group
909       \string"glslnumbers\string"^^J\space\space\space
910       :prefixes (\string"0\string" \string"1\string"
911       \string"2\string" \string"3\string" \string"4\string"
912       \string"5\string" \string"6\string" \string"7\string"
913       \string"8\string" \string"9\string")^^J\space\space\space
914       :before \string"@glslfirstletter\string")}
915   \else
916     \def\@xdylettergroups{}
917   \fi
918 \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.

```

919 \newcommand*\GlsAddLetterGroup[2]{%
920   \expandafter\toks@\expandafter{\@xdylettergroups}%
921   \protected@edef\@xdylettergroups{\the\toks@^^J%
922     (define-letter-group \string"#1\string"^^J\space\space\space#2)}%
923 }%

```

## 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.

```

924 \newcommand*\forallglossaries}[3][\@glo@types]{%
925   \@for#2:=#1\do{\ifx#2\@empty\else#3\fi}%
926 }

```

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

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

where  $\langle type \rangle$  is the glossary label and  $\langle cmd \rangle$  is a control sequence which will be set to the entry label in the current iteration.

```
927 \newcommand*{\forlgsentries}[3][\glsdefaulttype]{%
928   \edef\@glo@list{\csname glolist@#1\endcsname}%
929   \@for#2:=\@glo@list\do{\ifx#2\@empty\else#3\fi}%
930 }
```

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

```
\foralllgsentries[ $\langle glossary list \rangle$ ]{ $\langle cmd \rangle$ }{ $\langle code \rangle$ }
```

Within `\foralllgsentries`, the current glossary type is given by `\@thisglo@`.

```
931 \newcommand*{\foralllgsentries}[3][\@glo@types]{%
932 \expandafter\foralllglossaries\expandafter[#1]{\@thisglo@}{%
933 \forlgsentries[\@thisglo@]{#2}{#3}}}
```

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

```
\ifglossaryexists{ $\langle type \rangle$ }{ $\langle true-text \rangle$ }{ $\langle false-text \rangle$ }
```

where  $\langle type \rangle$  is the glossary's label.

```
934 \newcommand{\ifglossaryexists}[3]{%
935   \ifcsundef{glo@#1@out}{#3}{#2}%
936 }
```

**\iflgsentryexists** To check to see if a glossary entry has been defined use:

```
\iflgsentryexists{ $\langle label \rangle$ }{ $\langle true text \rangle$ }{ $\langle false text \rangle$ }
```

where  $\langle label \rangle$  is the entry's label.

```
937 \newcommand{\iflgsentryexists}[3]{%
938   \ifcsundef{glo@#1@name}{#3}{#2}%
939 }
```

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

```
\ifglsused{ $\langle label \rangle$ }{ $\langle true text \rangle$ }{ $\langle false text \rangle$ }
```

where  $\langle label \rangle$  is the entry's label. If true it will do  $\langle true text \rangle$  otherwise it will do  $\langle false text \rangle$ .

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

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

```
\glsdoifexists \glsdoifexists{ $\langle label \rangle$ }{ $\langle code \rangle$ }
```

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

```

941 \newcommand{\glsdoifexists}[2]{%
942   \ifglstryexists{#1}{#2}{%
943     \PackageError{glossaries}{Glossary entry ‘#1’ has not been
944     defined}{You need to define a glossary entry before you
945     can use it.}}%
946 }
```

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

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

```

947 \newcommand{\glsdoifnoexists}[2]{%
948   \ifglstryexists{#1}{%
949     \PackageError{glossaries}{Glossary entry ‘#1’ has already
950     been defined}{}}{#2}%
951 }
```

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

```

952 \newcommand{\ifglshaschildren}[3]{%
953   \glsdoifexists{#1}%
954   {%
955     \def\do@glshaschildren{#3}%
956     \expandafter\forglstryentries\expandafter[\csname glo@#1@type\endcsname]
957     {\glo@label}%
958     {%
959       \letcs\glo@parent{glo@\glo@label @parent}%
960       \ifthenelse{\equal{#1}{\glo@parent}}{%
961         {%
962           \def\do@glshaschildren{#2}%
963           \@endfortrue
964         }%
965         {}%
966       }%
967       \do@glshaschildren
968     }%
969 }
```

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

```

970 \newcommand{\ifglshasparent}[3]{%
971   \glsdoifexists{#1}%
972   {%
973     \ifcsempy{glo@#1@parent}{#3}{#2}%
974   }%
975 }
```

## 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`

```
976 \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`

```
977 \newcommand*{\newglossary}[5][glg]{%
978 \ifglossaryexists{#2}{%
979   \PackageError{glossaries}{Glossary type ‘#2’ already exists}{%
980     You can’t define a new glossary called ‘#2’ because it already
981     exists}%
982 }{%
```

Check if default has been set

```
983   \ifx\glsdefaulttype\relax
984     \gdef\glsdefaulttype{#2}%
985   \fi
```

Add this to the list of glossary types:

```
986   \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.

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

Store details of this new glossary type:

```
988   \expandafter\def\csname @glotype@#2@in\endcsname{#3}%
989   \expandafter\def\csname @glotype@#2@out\endcsname{#4}%
990   \expandafter\def\csname @glotype@#2@title\endcsname{#5}%
991   \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.

```

992 \ifcsundef{gls@#2@display}%
993 {%
994   \expandafter\gdef\csname gls@#2@display\endcsname{\glsdisplay}%
995 }%
996 {}%
997 \ifcsundef{gls@#2@displayfirst}%
998 {%
999   \expandafter\gdef\csname gls@#2@displayfirst\endcsname{%
1000     \glsdisplayfirst
1001   }%
1002 }%
1003 {}%
```

Define sort counter if required:

```

1004 \@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.)

```

1005 \@ifnextchar[{\@gls@setcounter{#2}}%
1006   {\@gls@setcounter{#2}[\glscounter]}}%
1007 }
```

`\altnewglossary`

```

1008 \newcommand*\altnewglossary}[3]{%
1009   \newglossary[#2-glg]{#1}{#2-gls}{#2-glo}{#3}%
1010 }
```

Only define new glossaries in the preamble:

```

1011 \@onlypreamble{\newglossary}
```

Only define new glossaries before `\makeglossaries`

```

1012 \@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  $\text{\LaTeX}$ , `\@newglossary` simply ignores its arguments.

`\@newglossary`

```

1013 \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`

```

1014 \def\@gls@setcounter#1[#2]{%
1015   \expandafter\def\csname @glsotype@#1@counter\endcsname{#2}%
```

Add counter to xindy list, if not already added:

```
1016 \ifglsxindy
1017   \GlsAddXdyCounters{#2}%
1018 \fi
1019 }
```

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

`\@gls@getcounter`

```
1020 \newcommand*{\@gls@getcounter}[1]{%
1021 \csname @glotype@#1@counter\endcsname}
```

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

```
1022 \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.

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

**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.)

```
1026 \define@key{glossentry}{description}{%
1027 \def\@glo@desc{#1}%
1028 }
```

descriptionplural

```
1029 \define@key{glossentry}{descriptionplural}{%  
1030 \def\@glo@descplural{#1}%  
1031 }
```

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

```
1032 \define@key{glossentry}{sort}{%  
1033 \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.

```
1034 \define@key{glossentry}{text}{%  
1035 \def\@glo@text{#1}%  
1036 }
```

**plural** The plural key determines how the plural form of the term should be displayed in the document. If omitted, the plural is constructed by appending `\glspluralsuffix` to the value of the text key.

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

**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.

```
1040 \define@key{glossentry}{first}{%  
1041 \def\@glo@first{#1}%  
1042 }
```

**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.

```
1043 \define@key{glossentry}{firstplural}{%  
1044 \def\@glo@firstplural{#1}%  
1045 }
```

**symbol** The symbol key is ignored by most of the predefined glossary styles, and defaults to `\relax` if omitted. It is provided for glossary styles that require an associated symbol, as well as a name and description. To make this value appear in the glossary, you need to redefine `\glossaryentryfield` so that it uses its fourth parameter. If you want this value to appear in the text when the term is used by commands like `\gls`, you will need to change `\glsdisplay` and

`\glsdisplayfirst` (either explicitly for all glossaries or via `\defglsdisplay` and `\defglsdisplayfirst` for individual glossaries).

```
1046 \define@key{glossentry}{symbol}{%
1047 \def\@glo@symbol{#1}%
1048 }
```

`symbolplural`

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

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

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

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

```
1054 \define@key{glossentry}{counter}{%
1055 \ifcsundef{c@#1}%
1056 {%
1057 \PackageError{glossaries}%
1058 {There is no counter called ‘#1’}%
1059 {%
1060 The counter key should have the name of a valid counter
1061 as its value%
1062 }%
1063 }%
1064 {%
1065 \def\@glo@counter{#1}%
1066 }%
1067 }
```

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

```
1068 \define@key{glossentry}{see}{%
1069 \def\@glo@see{#1}%
1070 \@glo@seeautonumberlist
1071 }
```

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

```
1072 \define@key{glossentry}{parent}{%
1073 \def\@glo@parent{#1}}
```

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

```
1074 \define@choicekey{glossentry}{nonumberlist}[\val\nr]{true,false}[true]{%
1075 \ifcase\nr\relax
1076 \def\@glo@prefix{\glsnonextpages}%
1077 }
```

```

1077 \else
1078 \def\@glo@prefix{\glsnextpages}%
1079 \fi
1080 }

```

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

user1

```

1081 \define@key{glossentry}{user1}{%
1082 \def\@glo@useri{#1}%
1083 }

```

user2

```

1084 \define@key{glossentry}{user2}{%
1085 \def\@glo@userii{#1}%
1086 }

```

user3

```

1087 \define@key{glossentry}{user3}{%
1088 \def\@glo@useriii{#1}%
1089 }

```

user4

```

1090 \define@key{glossentry}{user4}{%
1091 \def\@glo@useriv{#1}%
1092 }

```

user5

```

1093 \define@key{glossentry}{user5}{%
1094 \def\@glo@userv{#1}%
1095 }

```

user6

```

1096 \define@key{glossentry}{user6}{%
1097 \def\@glo@uservi{#1}%
1098 }

```

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.

```

1099 \define@key{glossentry}{short}{%
1100 \def\@glo@short{#1}%
1101 }

```

shortplural This key is provided for use by \newacronym.

```

1102 \define@key{glossentry}{shortplural}{%
1103 \def\@glo@shortpl{#1}%
1104 }

```

long This key is provided for use by \newacronym.

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

longplural This key is provided for use by \newacronym.

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

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

```
1111 \newcommand*{\@glsnoname}{%
1112   \PackageError{glossaries}{name key required in
1113   \string\newglossaryentry\space for entry '\@glo@label'}{You
1114   haven't specified the entry name}}
```

\@glsdefaultplural Define command to set default plural.

```
1115 \newcommand*{\@glsdefaultplural}{\@glo@text\glspluralsuffix}
```

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

```
1116 \newcommand*{\@gls@missingnumberlist}[1]{%
1117   ??%
1118   \ifglssavenumberlist
1119     \GlossariesWarning{Missing number list for entry '#1'.
1120     Maybe makeglossaries + rerun required.}%
1121   \else
1122     \PackageError{glossaries}%
1123     {Package option 'savenumberlist=true' required.}%
1124     {%
1125     You must use the 'savenumberlist' package option
1126     to reference location lists.%
1127     }%
1128   \fi
1129 }
```

\@glsdefaultsort Define command to set default sort.

```
1130 \newcommand*{\@glsdefaultsort}{\@glo@name}
```

\gls@level Register to increment entry levels.

```
1131 \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.)

```
1132 \newrobustcmd{\newglossaryentry}[2]{%
```

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

```
1133   \glsdoifnoexists{#1}%
1134   {%
```

Store label

```
1135 \def\@glo@label{#1}%
```

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

```
1136 \let\@glo@name\@glsnoname
```

```
1137 \def\@glo@desc{%
```

```
1138 \PackageError{glossaries}
```

```
1139 {%
```

```
1140 description key required in \string\newglossaryentry\space
```

```
1141 for entry '\@glo@label'%
```

```
1142 }%
```

```
1143 {%
```

```
1144 You haven't specified the entry description%
```

```
1145 }%
```

```
1146 }%
```

```
1147 \def\@glo@descplural{\@glo@desc}%
```

```
1148 \def\@glo@type{\glsdefaulttype}%
```

```
1149 \def\@glo@symbol{\relax}%
```

```
1150 \def\@glo@symbolplural{\@glo@symbol}%
```

```
1151 \def\@glo@text{\@glo@name}%
```

```
1152 \let\@glo@plural\@glsdefaultplural
```

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

```
1153 \let\@glo@first\relax
```

```
1154 \let\@glo@firstplural\relax
```

Set the default sort:

```
1155 \let\@glo@sort\@glsdefaultsort
```

Set the default counter:

```
1156 \def\@glo@counter{\@gls@getcounter{\@glo@type}}%
```

```
1157 \def\@glo@see{}%
```

```
1158 \def\@glo@parent{}%
```

```
1159 \def\@glo@prefix{}%
```

```
1160 \def\@glo@useri{}%
```

```
1161 \def\@glo@userii{}%
```

```
1162 \def\@glo@useriii{}%
```

```
1163 \def\@glo@useriv{}%
```

```
1164 \def\@glo@userv{}%
```

```
1165 \def\@glo@uservi{}%
```

```

1166 \def\@glo@short{}%
1167 \def\@glo@shortpl{}%
1168 \def\@glo@long{}%
1169 \def\@glo@longpl{}%

Add start hook in case another package wants to add extra keys.
1170 \@newglossaryentryprehook

Extract key-val information from third parameter:
1171 \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.
1172 \ifcsundef{glolist@\@glo@type}%
1173 {%
1174 \PackageError{glossaries}%
1175 {Glossary type '\@glo@type' has not been defined}%
1176 {You need to define a new glossary type, before making entries
1177 in it}%
1178 }%
1179 {%
1180 \protected@edef\@glolist@{\csname glolist@\@glo@type\endcsname}%
1181 \expandafter\xdef\csname glolist@\@glo@type\endcsname{\@glolist@{#1},}%
1182 }%

Initialise level to 0.
1183 \gls@level=0\relax

Has this entry been assigned a parent?
1184 \ifx\@glo@parent\@empty

Doesn't have a parent. Set \glo@<label>@parent to empty.
1185 \expandafter\gdef\csname glo@#1@parent\endcsname{}%
1186 \else

Has a parent. Check to ensure this entry isn't its own parent.
1187 \ifthenelse{\equal{#1}{\@glo@parent}}{%
1188 {%
1189 \PackageError{glossaries}{Entry '#1' can't be its own parent}{}%
1190 \def\@glo@parent{}%
1191 \expandafter\gdef\csname glo@#1@parent\endcsname{}%
1192 }%
1193 {%

Check the parent exists:
1194 \ifglsentryexists{\@glo@parent}%
1195 {%

Parent exists. Set \glo@<label>@parent.
1196 \expandafter\xdef\csname glo@#1@parent\endcsname{\@glo@parent}%

Determine level.
1197 \gls@level=\csname glo@\@glo@parent @level\endcsname\relax
1198 \advance\gls@level by 1\relax

```

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

```

1199      \ifx\@glo@name\@glsnoname
1200      \expandafter\let\expandafter\@glo@name
1201      \csname glo@\@glo@parent @name\endcsname

```

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

```

1202      \ifx\@glo@plural\@glsdefaultplural
1203      \expandafter\let\expandafter\@glo@plural
1204      \csname glo@\@glo@parent @plural\endcsname
1205      \fi
1206      \fi
1207      }%
1208      {%

```

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

```

1209      \PackageError{glossaries}%
1210      {%
1211      Invalid parent '\@glo@parent'
1212      for entry '#1' - parent doesn't exist%
1213      }%
1214      {%
1215      Parent entries must be defined before their children%
1216      }%
1217      \def\@glo@parent{%
1218      \expandafter\gdef\csname glo@#1@parent\endcsname{%
1219      }%
1220      }%
1221      \fi

```

Set the level for this entry

```

1222      \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.)

```

1223      \ifx\relax\@glo@firstplural
1224      \ifx\relax\@glo@first
1225      \def\@glo@firstplural{\@glo@plural}%
1226      \def\@glo@first{\@glo@text}%
1227      \else
1228      \def\@glo@firstplural{\@glo@first\glspluralsuffix}%
1229      \fi
1230      \else
1231      \ifx\relax\@glo@first
1232      \def\@glo@first{\@glo@text}%
1233      \fi
1234      \fi

```

Define commands associated with this entry:

```

1235 \expandafter
1236 \protected@xdef\csname glo@#1@text\endcsname{\@glo@text}%
1237 \expandafter
1238 \protected@xdef\csname glo@#1@plural\endcsname{\@glo@plural}%
1239 \expandafter
1240 \protected@xdef\csname glo@#1@first\endcsname{\@glo@first}%
1241 \expandafter
1242 \protected@xdef\csname glo@#1@firstpl\endcsname{\@glo@firstplural}%
1243 \expandafter
1244 \protected@xdef\csname glo@#1@type\endcsname{\@glo@type}%
1245 \expandafter
1246 \protected@xdef\csname glo@#1@counter\endcsname{\@glo@counter}%
1247 \expandafter
1248 \protected@xdef\csname glo@#1@useri\endcsname{\@glo@useri}%
1249 \expandafter
1250 \protected@xdef\csname glo@#1@userii\endcsname{\@glo@userii}%
1251 \expandafter
1252 \protected@xdef\csname glo@#1@useriii\endcsname{\@glo@useriii}%
1253 \expandafter
1254 \protected@xdef\csname glo@#1@useriv\endcsname{\@glo@useriv}%
1255 \expandafter
1256 \protected@xdef\csname glo@#1@userv\endcsname{\@glo@userv}%
1257 \expandafter
1258 \protected@xdef\csname glo@#1@uservi\endcsname{\@glo@uservi}%
1259 \expandafter
1260 \protected@xdef\csname glo@#1@short\endcsname{\@glo@short}%
1261 \expandafter
1262 \protected@xdef\csname glo@#1@shortpl\endcsname{\@glo@shortpl}%
1263 \expandafter
1264 \protected@xdef\csname glo@#1@long\endcsname{\@glo@long}%
1265 \expandafter
1266 \protected@xdef\csname glo@#1@longpl\endcsname{\@glo@longpl}%
1267 \@gls@sanitizename
1268 \expandafter\protected@xdef\csname glo@#1@name\endcsname{\@glo@name}%

```

Set default numberlist if not defined:

```

1269 \ifcsundef{glo@#1@numberlist}%
1270 {%
1271 \csxdef{glo@#1@numberlist}{\noexpand\@gls@missingnumberlist{\@glo@label}}%
1272 }%
1273 {}%

```

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.

```

1274 \def\@glo@@desc{\@glo@first}%
1275 \ifx\@glo@desc\@glo@@desc
1276 \let\@glo@desc\@glo@first
1277 \fi

```

```

1278 \@gls@sanitizedesc
1279 \expandafter\protected@xdef\csname glo@#1@desc\endcsname{\@glo@desc}%
1280 \expandafter\protected@xdef\csname glo@#1@descplural\endcsname{\@glo@descplural}%

```

Set the sort key for this entry:

```

1281 \@gls@defsort{\@glo@type}{#1}%

1282 \def\@glo@@symbol{\@glo@text}%
1283 \ifx\@glo@symbol\@glo@@symbol
1284 \let\@glo@symbol\@glo@text
1285 \fi
1286 \@gls@sanitizesymbol
1287 \expandafter\protected@xdef\csname glo@#1@symbol\endcsname{\@glo@symbol}%
1288 \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):

```

1289 \expandafter\gdef\csname glo@#1@flagfalse\endcsname{%
1290 \expandafter\global\expandafter
1291 \let\csname ifglo@#1@flag\endcsname\iffalse
1292 }%
1293 \expandafter\gdef\csname glo@#1@flagtrue\endcsname{%
1294 \expandafter\global\expandafter
1295 \let\csname ifglo@#1@flag\endcsname\iftrue
1296 }%
1297 \csname glo@#1@flagfalse\endcsname

```

Sort out any cross-referencing if required.

```

1298 \ifx\@glo@see\@empty
1299 \else
1300 \protected@edef\@do@glsee{%
1301 \noexpand\@gls@fixbraces\noexpand\@glo@list\@glo@see
1302 \noexpand\@nil
1303 \noexpand\expandafter\noexpand\@glsee\noexpand\@glo@list{#1}}%
1304 \@do@glsee
1305 \fi
1306 }%

```

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

```

1307 \do@glo@storeentry{#1}%

```

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

```

1308 \@newglossaryentryposthook
1309 }

```

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

```

1310 \newcommand*{\@newglossaryentryprehook}{}

```

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

```

1311 \newcommand*{\@newglossaryentryposthook}{}

```

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

```

1312 \newcommand*{\glsmoveentry}[2]{%
1313   \edef\glo@type{\csname glo@#1@type\endcsname}%
1314   \def\glo@list{,}%
1315   \forlslentries[\glo@type]{\glo@label}%
1316   {%
1317     \ifthenelse{\equal{\glo@label}{#1}}{\eappto\glo@list{\glo@label,}}%
1318   }%
1319   \cslet{glolist@\glo@type}{\glo@list}%
1320   \csdef{glo@#1@type}{#2}%
1321 }
```

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

```

1322 \ifglxindy
1323   \newcommand*{\@glossaryentryfield}{\string\@glossaryentryfield}
1324 \else
1325   \newcommand*{\@glossaryentryfield}{\string\glossaryentryfield}
1326 \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.)

```

1327 \ifglxindy
1328   \newcommand*{\@glossarysubentryfield}{%
1329     \string\@glossarysubentryfield}
1330 \else
1331   \newcommand*{\@glossarysubentryfield}{%
1332     \string\glossarysubentryfield}
1333 \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.)

```

1334 \newcommand{\@glo@storeentry}[1]{%
  Get the sort string and escape any special characters
1335 \protected@edef\@glo@sort{\csname glo@#1@sort\endcsname}%
1336 \@gls@checkmkidxchars\@glo@sort
```

Same again for the name string.

```

1337 \protected@edef\@glo@name{\csname glo@#1@name\endcsname}%
1338 \@gls@checkmkidxchars\@glo@name
```

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

```

1339 \ifglxindy
```

```

1340 \protected@edef\@glo@name{\string\@glsnamefont{\@glo@name}}%
1341 \else
1342 \protected@edef\@glo@name{\string\@glsnamefont{\@glo@name}}%
1343 \fi

```

Get the description string and escape any special characters

```

1344 \protected@edef\@glo@desc{\csname glo@#1@desc\endcsname}%
1345 \@gls@checkmkidxchars\@glo@desc

```

Same again for the symbol

```

1346 \protected@edef\@glo@symbol{\csname glo@#1@symbol\endcsname}%
1347 \@gls@checkmkidxchars\@glo@symbol

```

Escape any special characters in the prefix

```

1348 \@gls@checkmkidxchars\@glo@prefix

```

Get the parent, if one exists

```

1349 \edef\@glo@parent{\csname glo@#1@parent\endcsname}%

```

Write the information to the glossary file.

```

1350 \ifglxindy

```

Store using xindy syntax.

```

1351 \ifx\@glo@parent\@empty

```

Entry doesn't have a parent

```

1352 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
1353 (\string"\@glo@sort\string" %
1354 \string"\@glo@prefix\@glossaryentryfield{#1}{\@glo@name
1355 }\@glo@desc}{\@glo@symbol}\string") %
1356 }%
1357 \else

```

Entry has a parent

```

1358 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
1359 \csname glo@\@glo@parent @index\endcsname
1360 (\string"\@glo@sort\string" %
1361 \string"\@glo@prefix\@glossarysubentryfield%
1362 {\csname glo@#1@level\endcsname}{#1}{\@glo@name
1363 }\@glo@desc}{\@glo@symbol}\string") %
1364 }%
1365 \fi
1366 \else

```

Store using makeindex syntax.

```

1367 \ifx\@glo@parent\@empty

```

Sanitize \@glo@prefix

```

1368 \@onelevel@sanitize\@glo@prefix

```

Entry doesn't have a parent

```

1369 \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
1370 \@glo@sort\@gls@actualchar\@glo@prefix
1371 \@glossaryentryfield{#1}{\@glo@name}{\@glo@desc

```

```

1372     }\@glo@symbol}%
1373   }%
1374   \else
      Entry has a parent
1375   \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
1376     \csname glo@\@glo@parent @index\endcsname\@gls@levelchar
1377     \@glo@sort\@gls@actualchar\@glo@prefix
1378     \@glossarysubentryfield
1379     {\csname glo@#1@level\endcsname}{#1}{\@glo@name}{\@glo@desc
1380     }\@glo@symbol}%
1381   }%
1382   \fi
1383 \fi
1384 }

```

## 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`

```

1385 \newcommand*{\glsreset}[1]{%
1386 \glsdoifexists{#1}{%
1387 \expandafter\global\csname glo@#1@flagfalse\endcsname}}

```

As above, but with only a local effect:

`\glslocalreset`

```

1388 \newcommand*{\glslocalreset}[1]{%
1389 \glsdoifexists{#1}{%
1390 \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`

```

1391 \newcommand*{\glsunset}[1]{%
1392 \glsdoifexists{#1}{%
1393 \expandafter\global\csname glo@#1@flagtrue\endcsname}}

```

As above, but with only a local effect:

`\glslocalunset`

```

1394 \newcommand*{\glslocalunset}[1]{%
1395 \glsdoifexists{#1}{%
1396 \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`

```
1397 \newcommand*{\glsresetall}[1][\@glo@types]{%
1398 \forallglsentries[#1]{\@glsentry}{%
1399 \glsreset{\@glsentry}}}
```

As above, but with only a local effect:

`\glslocalresetall`

```
1400 \newcommand*{\glslocalresetall}[1][\@glo@types]{%
1401 \forallglsentries[#1]{\@glsentry}{%
1402 \glslocalreset{\@glsentry}}}
```

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

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

`\glsunsetall`

```
1403 \newcommand*{\glsunsetall}[1][\@glo@types]{%
1404 \forallglsentries[#1]{\@glsentry}{%
1405 \glsunset{\@glsentry}}}
```

As above, but with only a local effect:

`\glslocalunsetall`

```
1406 \newcommand*{\glslocalunsetall}[1][\@glo@types]{%
1407 \forallglsentries[#1]{\@glsentry}{%
1408 \glslocalunset{\@glsentry}}}
```

## 1.9 Loading files containing glossary entries

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

`\loadglsentries[⟨type⟩]{⟨filename⟩}`

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

`\loadglsentries`

```
1409 \newcommand*{\loadglsentries}[2][\@gls@default]{%
1410 \let\@gls@default\glsdefaulttype
1411 \def\glsdefaulttype{#1}\input{#2}%
1412 \let\glsdefaulttype\@gls@default}
```

---

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

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

```
1413 \onlypreamble{\loadglsentries}
```

## 1.10 Using glossary entries in the text

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

`\glstextformat`

```
1414 \newcommand*{\glstextformat}[1]{#1}
```

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

`\glsdisplayfirst`

```
1415 \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`

```
1416 \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`

```

1417 \newcommand*\defglsdisplay}[2][\glsdefaulttype]{%
1418 \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
```

```

1419 \newcommand*\defglsdisplayfirst}[2][\glsdefaulttype]{%
1420 \expandafter\def\csname gls@#1@displayfirst\endcsname##1##2##3##4{#2}}

```

### 1.10.1 Links to glossary entries

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

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

```

1421 \define@key{glslink}{counter}{%
1422   \ifcsundef{c@#1}%
1423   {%
1424     \PackageError{glossaries}%
1425     {There is no counter called ‘#1’}%
1426     {%
1427       The counter key should have the name of a valid counter
1428       as its value%
1429     }%
1430   }%
1431   {%
1432     \def\@gls@counter{#1}%
1433   }%
1434 }

```

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.

```

1435 \define@key{glslink}{format}{%
1436 \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.

```
1437 \define@boolkey{glslink}{hyper}[true]{}

```

The local key is a boolean key. If true this indicates that commands such as \gls should only do a local reset rather than a global one.

```
1438 \define@boolkey{glslink}{local}[true]{}

```

Syntax:

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

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

There is also a starred version:

```
\glslink*[⟨options⟩]{⟨label⟩}{⟨text⟩}
```

which is equivalent to \glslink[hyper=false,⟨options⟩]{⟨label⟩}{⟨text⟩}

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

```
\glslink

```

```
1439 \newrobustcmd*{\glslink}{%
1440 \@ifstar\@sgls@link\@gls@link}
```

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

```
1441 \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.

```
1442 \newcommand*{\@gls@link}[3] [] {%
1443   \ifglstryexists{#2}%
1444   {%
1445     \@gls@link[#1]{#2}{#3}%
1446   }{%
1447     \PackageError{glossaries}{Glossary entry ‘#2’ has not been
1448     defined}{You need to define a glossary entry before you
1449     can use it.}%

```

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

```
1450     \glstextformat{#3}%
1451   }%
1452 }
```

`\@gls@link`

```
1453 \def\@gls@link[#1]#2#3{%
```

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

```
1454 \leavevmode
```

```
1455 \def\glslabel{#2}%
```

```
1456 \def\@glsnumberformat{glsnumberformat}%
```

```
1457 \edef\@gls@counter{\csname glo@#2@counter\endcsname}%
```

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

```
1458 \expandafter\xifinlist\expandafter
```

```
1459 {\csname glo@#2@type\endcsname}\@gls@nohyperlist}%
```

```
1460 {%
```

```
1461 \KV@glslink@hyperfalse
```

```
1462 }%
```

```
1463 {%
```

```
1464 \KV@glslink@hypertrue
```

```
1465 }%
```

```
1466 \setkeys{glslink}{#1}%
```

Store the entry’s counter in `\theglsentrycounter`

```
1467 \@gls@saveentrycounter
```

Define sort key if necessary:

```
1468 \@gls@setsort{#2}%
```

```
1469 \@do@wrglossary{#2}%
```

```
1470 \ifKV@glslink@hyper
```

```
1471 \@glslink{\glolinkprefix#2}{\glstextformat{#3}}%
```

```
1472 \else
```

```
1473 \glstextformat{#3}\relax
```

```
1474 \fi
```

```
1475 }
```

`\glolinkprefix`

```
1476 \newcommand*{\glolinkprefix}{glo:}
```

`\glsentrycounter` Set default value of entry counter

```
1477 \def\glsentrycounter{\glscounter}%
```

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

```
1478 \newcommand*{\@gls@saveentrycounter}{%
```

```
1479 \def\@gls@Hcounter{}%
```

Are we using equation counter?

```
1480 \ifthenelse{\equal{\@gls@counter}{equation}}%
```

```
1481 {
```

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

```

1482 \ifcsundef{xatlevel@}%
1483 {%
1484 \edef\theglentrycounter{\expandafter\noexpand
1485 \csname the\@gls@counter\endcsname}%
1486 }%
1487 {%
1488 \ifx\xatlevel@\@empty
1489 \edef\theglentrycounter{\expandafter\noexpand
1490 \csname the\@gls@counter\endcsname}%
1491 \else
1492 \savecounters@
1493 \advance\c@equation by 1\relax
1494 \edef\theglentrycounter{\csname the\@gls@counter\endcsname}%

```

Check if hyperref version of this counter

```

1495 \ifcsundef{theH\@gls@counter}%
1496 {%
1497 \def\@gls@Hcounter{\theglentrycounter}%
1498 }%
1499 {%
1500 \def\@gls@Hcounter{\csname theH\@gls@counter\endcsname}%
1501 }%
1502 \protected@edef\theHglentrycounter{\@gls@Hcounter}%
1503 \restorecounters@
1504 \fi
1505 }%
1506 }%
1507 {%

```

Not using equation counter so no special measures:

```

1508 \edef\theglentrycounter{\expandafter\noexpand
1509 \csname the\@gls@counter\endcsname}%
1510 }%

```

Check if hyperref version of this counter

```

1511 \ifx\@gls@Hcounter\@empty
1512 \ifcsundef{theH\@gls@counter}%
1513 {%
1514 \def\theHglentrycounter{\theglentrycounter}%
1515 }%
1516 {%
1517 \protected@edef\theHglentrycounter{\expandafter\noexpand
1518 \csname theH\@gls@counter\endcsname}%
1519 }%
1520 \fi
1521 }

```

`\@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.

```

1522 \def\@set@glo@numformat#1#2#3#4{%
1523   \expandafter\@glo@check@mkidxrangechar#3\@nil
1524   \protected@edef#1{%
1525     \@glo@prefix setentrycounter[#4]{#2}%
1526     \expandafter\string\csname\@glo@suffix\endcsname
1527   }%
1528   \@gls@checkmkidxchars#1%
1529 }

```

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.

```

1530 \def\@glo@check@mkidxrangechar#1#2\@nil{%
1531   \if#1(\relax
1532     \def\@glo@prefix{}%
1533     \if\relax#2\relax
1534       \def\@glo@suffix{glsnumberformat}%
1535     \else
1536       \def\@glo@suffix{#2}%
1537     \fi
1538   \else
1539     \if#1)\relax
1540       \def\@glo@prefix{)}%
1541     \if\relax#2\relax
1542       \def\@glo@suffix{glsnumberformat}%
1543     \else
1544       \def\@glo@suffix{#2}%
1545     \fi
1546   \else
1547     \def\@glo@prefix{}\def\@glo@suffix{#1#2}%
1548   \fi
1549 }

```

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

```

1550 \newcommand*\@gls@escbsdq[1]{%
1551   \def\@gls@checkedmkidx{%
1552     \let\gls@xdystring=#1\relax
1553     \@onelevel@sanitize\gls@xdystring
1554     \edef\do@gls@xdycheckbackslash{%
1555       \noexpand\@gls@xdycheckbackslash\gls@xdystring\noexpand\@nil
1556       \@backslashchar\@backslashchar\noexpand\null}%

```

```

1557 \do@gl@xdycheckbackslash
1558 \expandafter\@gl@updatechecked\@gl@checkedmkidx{\gl@xdystring}%
1559 \def\@gl@checkedmkidx{}%
1560 \expandafter\@gl@xdycheckquote\gl@xdystring\@nil""\null
1561 \expandafter\@gl@updatechecked\@gl@checkedmkidx{\gl@xdystring}%

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

1562 \@for\@gl@tmp:=\gl@protected@pagefmts\do
1563 {%
1564 \edef\@gl@sanitized@tmp{\expandafter\@gobble\string\\expandonce\@gl@tmp}%
1565 \@onelevel@sanitize\@gl@sanitized@tmp
1566 \edef\gl@dostubst{%
1567 \noexpand\DTLsubstituteall\noexpand\gl@xdystring
1568 {\@gl@sanitized@tmp}{\expandonce\@gl@tmp}%
1569 }%
1570 \gl@dostubst
1571 }%

Assign to required control sequence
1572 \let#1=\gl@xdystring
1573 }

```

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

gl@checkmkidxchars

```

1574 \newcommand{\@gl@checkmkidxchars}[1]{%
1575 \ifgl@xindy
1576 \@gl@escbsdq{#1}%
1577 \else
1578 \def\@gl@checkedmkidx{}%
1579 \expandafter\@gl@checkquote#1\@nil""\null
1580 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
1581 \def\@gl@checkedmkidx{}%
1582 \expandafter\@gl@checkescquote#1\@nil\""\null
1583 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
1584 \def\@gl@checkedmkidx{}%
1585 \expandafter\@gl@checkescactual#1\@nil\?\?\null
1586 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
1587 \def\@gl@checkedmkidx{}%
1588 \expandafter\@gl@checkactual#1\@nil??\null
1589 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
1590 \def\@gl@checkedmkidx{}%
1591 \expandafter\@gl@checkbar#1\@nil||\null
1592 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
1593 \def\@gl@checkedmkidx{}%
1594 \expandafter\@gl@checkesbar#1\@nil\\|\null
1595 \expandafter\@gl@updatechecked\@gl@checkedmkidx{#1}%
1596 \def\@gl@checkedmkidx{}%
1597 \expandafter\@gl@checklevel#1\@nil!!\null

```

```

1598 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
1599 \fi
1600 }

```

Update the control sequence and strip trailing \@nil:

\@gls@updatechecked

```

1601 \def\@gls@updatechecked#1\@nil#2{\def#2{#1}}

```

\@gls@tmpb Define temporary token

```

1602 \newtoks\@gls@tmpb

```

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

```

1603 \def\@gls@checkquote#1"#2"#3\null{%
1604 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1605 \toks@={#1}%
1606 \ifx\null#2\null
1607 \ifx\null#3\null
1608 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1609 \def\@gls@checkquote{\relax}%
1610 \else
1611 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1612 \@gls@quotechar\@gls@quotechar\@gls@quotechar\@gls@quotechar}%
1613 \def\@gls@checkquote{\@gls@checkquote#3\null}%
1614 \fi
1615 \else
1616 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1617 \@gls@quotechar\@gls@quotechar}%
1618 \ifx\null#3\null
1619 \def\@gls@checkquote{\@gls@checkquote#2""\null}%
1620 \else
1621 \def\@gls@checkquote{\@gls@checkquote#2"#3\null}%
1622 \fi
1623 \fi
1624 \@gls@checkquote}

```

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

```

1625 \def\@gls@checkescquote#1\"#2\"#3\null{%
1626 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1627 \toks@={#1}%
1628 \ifx\null#2\null
1629 \ifx\null#3\null
1630 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1631 \def\@gls@checkescquote{\relax}%
1632 \else
1633 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1634 \@gls@quotechar\string\" \@gls@quotechar
1635 \@gls@quotechar\string\" \@gls@quotechar}%
1636 \def\@gls@checkescquote{\@gls@checkescquote#3\null}%

```

```

1637 \fi
1638 \else
1639 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1640   \@gls@quotechar\string"\@gls@quotechar}%
1641 \ifx\null#3\null
1642   \def\@gls@checkescquote{\@gls@checkescquote#2\""\null}%
1643 \else
1644   \def\@gls@checkescquote{\@gls@checkescquote#2\"#3\null}%
1645 \fi
1646 \fi
1647 \@gls@checkescquote}

```

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

```

1648 \def\@gls@checkescactual#1\?#2\?#3\null{%
1649 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1650 \toks@={#1}%
1651 \ifx\null#2\null
1652 \ifx\null#3\null
1653   \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1654   \def\@gls@checkescactual{\relax}%
1655 \else
1656   \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1657     \@gls@quotechar\string"\@gls@actualchar
1658     \@gls@quotechar\string"\@gls@actualchar}%
1659   \def\@gls@checkescactual{\@gls@checkescactual#3\null}%
1660 \fi
1661 \else
1662   \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1663     \@gls@quotechar\string"\@gls@actualchar}%
1664   \ifx\null#3\null
1665     \def\@gls@checkescactual{\@gls@checkescactual#2\?\?\null}%
1666   \else
1667     \def\@gls@checkescactual{\@gls@checkescactual#2\?#3\null}%
1668   \fi
1669 \fi
1670 \@gls@checkescactual}

```

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

```

1671 \def\@gls@checkescbar#1\|#2\|#3\null{%
1672 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1673 \toks@={#1}%
1674 \ifx\null#2\null
1675 \ifx\null#3\null
1676   \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1677   \def\@gls@checkescbar{\relax}%
1678 \else
1679   \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1680     \@gls@quotechar\string"\@gls@encapchar
1681     \@gls@quotechar\string"\@gls@encapchar}%

```

```

1682 \def\@gls@checkescbar{\@gls@checkescbar#3\null}%
1683 \fi
1684 \else
1685 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1686 \@gls@quotechar\string"\@gls@encapchar}%
1687 \ifx\null#3\null
1688 \def\@gls@checkescbar{\@gls@checkescbar#2\\\null}%
1689 \else
1690 \def\@gls@checkescbar{\@gls@checkescbar#2|#3\null}%
1691 \fi
1692 \fi
1693 \@gls@checkescbar}

```

\@gls@checkesclevel Similarly for \!:

```

1694 \def\@gls@checkesclevel#1\!#2\!#3\null{%
1695 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1696 \toks@={#1}%
1697 \ifx\null#2\null
1698 \ifx\null#3\null
1699 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1700 \def\@gls@checkesclevel{\relax}%
1701 \else
1702 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1703 \@gls@quotechar\string"\@gls@levelchar
1704 \@gls@quotechar\string"\@gls@levelchar}%
1705 \def\@gls@checkesclevel{\@gls@checkesclevel#3\null}%
1706 \fi
1707 \else
1708 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1709 \@gls@quotechar\string"\@gls@levelchar}%
1710 \ifx\null#3\null
1711 \def\@gls@checkesclevel{\@gls@checkesclevel#2\!\!\null}%
1712 \else
1713 \def\@gls@checkesclevel{\@gls@checkesclevel#2\!#3\null}%
1714 \fi
1715 \fi
1716 \@gls@checkesclevel}

```

\@gls@checkbar and for |:

```

1717 \def\@gls@checkbar#1|#2|#3\null{%
1718 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1719 \toks@={#1}%
1720 \ifx\null#2\null
1721 \ifx\null#3\null
1722 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1723 \def\@gls@checkbar{\relax}%
1724 \else
1725 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1726 \@gls@quotechar\@gls@encapchar\@gls@quotechar\@gls@encapchar}%

```

```

1727 \def\@gls@checkbar{\@gls@checkbar#3\null}%
1728 \fi
1729 \else
1730 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1731 \@gls@quotechar\@gls@encapchar}%
1732 \ifx\null#3\null
1733 \def\@gls@checkbar{\@gls@checkbar#2||\null}%
1734 \else
1735 \def\@gls@checkbar{\@gls@checkbar#2|#3\null}%
1736 \fi
1737 \fi
1738 \@gls@checkbar}

```

\@gls@checklevel and for !:

```

1739 \def\@gls@checklevel#1!#2!#3\null{%
1740 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1741 \toks@={#1}%
1742 \ifx\null#2\null
1743 \ifx\null#3\null
1744 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1745 \def\@gls@checklevel{\relax}%
1746 \else
1747 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1748 \@gls@quotechar\@gls@levelchar\@gls@quotechar\@gls@levelchar}%
1749 \def\@gls@checklevel{\@gls@checklevel#3\null}%
1750 \fi
1751 \else
1752 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1753 \@gls@quotechar\@gls@levelchar}%
1754 \ifx\null#3\null
1755 \def\@gls@checklevel{\@gls@checklevel#2!!\null}%
1756 \else
1757 \def\@gls@checklevel{\@gls@checklevel#2!#3\null}%
1758 \fi
1759 \fi
1760 \@gls@checklevel}

```

\@gls@checkactual and for ?:

```

1761 \def\@gls@checkactual#1?#2?#3\null{%
1762 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1763 \toks@={#1}%
1764 \ifx\null#2\null
1765 \ifx\null#3\null
1766 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1767 \def\@gls@checkactual{\relax}%
1768 \else
1769 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1770 \@gls@quotechar\@gls@actualchar\@gls@quotechar\@gls@actualchar}%
1771 \def\@gls@checkactual{\@gls@checkactual#3\null}%

```

```

1772 \fi
1773 \else
1774 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1775   \@gls@quotechar\@gls@actualchar}%
1776 \ifx\null#3\null
1777   \def\@gls@checkactual{\@gls@checkactual#2??\null}%
1778 \else
1779   \def\@gls@checkactual{\@gls@checkactual#2?#3\null}%
1780 \fi
1781 \fi
1782 \@@gls@checkactual}

```

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

```

1783 \def\@gls@xdycheckquote#1"#2"#3\null{%
1784 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
1785 \toks@={#1}%
1786 \ifx\null#2\null
1787   \ifx\null#3\null
1788     \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
1789     \def\@gls@xdycheckquote{\relax}%
1790   \else
1791     \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1792       \string"\string"}%
1793     \def\@gls@xdycheckquote{\@gls@xdycheckquote#3\null}%
1794   \fi
1795 \else
1796 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
1797   \string"}%
1798 \ifx\null#3\null
1799   \def\@gls@xdycheckquote{\@gls@xdycheckquote#2""\null}%
1800 \else
1801   \def\@gls@xdycheckquote{\@gls@xdycheckquote#2"#3\null}%
1802 \fi
1803 \fi
1804 \@@gls@xdycheckquote
1805 }

```

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

```

\@gls@xdycheckbackslash
1806 \edef\def@gls@xdycheckbackslash{%
1807   \noexpand\def\noexpand\@gls@xdycheckbackslash##1\@backslashchar
1808     ##2\@backslashchar##3\noexpand\null{%
1809     \noexpand\@gls@tmpb=\noexpand\expandafter
1810       {\noexpand\@gls@checkedmkidx}%
1811     \noexpand\toks@={##1}%
1812     \noexpand\ifx\noexpand\null##2\noexpand\null
1813       \noexpand\ifx\noexpand\null##3\noexpand\null
1814         \noexpand\edef\noexpand\@gls@checkedmkidx{%
1815           \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@}%

```

```

1816 \noexpand\def\noexpand\@gls@xdycheckbackslash{\relax}%
1817 \noexpand\else
1818 \noexpand\edef\noexpand\@gls@checkedmkidx{%
1819 \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
1820 \@backslashchar\@backslashchar\@backslashchar\@backslashchar}%
1821 \noexpand\def\noexpand\@gls@xdycheckbackslash{%
1822 \noexpand\@gls@xdycheckbackslash##3\noexpand\null}%
1823 \noexpand\fi
1824 \noexpand\else
1825 \noexpand\edef\noexpand\@gls@checkedmkidx{%
1826 \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
1827 \@backslashchar\@backslashchar}%
1828 \noexpand\ifx\noexpand\null##3\noexpand\null
1829 \noexpand\def\noexpand\@gls@xdycheckbackslash{%
1830 \noexpand\@gls@xdycheckbackslash##2\@backslashchar
1831 \@backslashchar\noexpand\null}%
1832 \noexpand\else
1833 \noexpand\def\noexpand\@gls@xdycheckbackslash{%
1834 \noexpand\@gls@xdycheckbackslash##2\@backslashchar
1835 ##3\noexpand\null}%
1836 \noexpand\fi
1837 \noexpand\fi
1838 \noexpand\@gls@xdycheckbackslash
1839 }%
1840 }

```

Now go ahead and define \gls@xdycheckbackslash

```

1841 \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.

```

1842 \ifcsundef{hyperlink}%
1843 {%
1844 \gdef@glslink#1#2{#2}%
1845 }%
1846 {%
1847 \gdef@glslink#1#2{\hyperlink{#1}{#2}}%
1848 }

```

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

```

1849 \newlength@gls@tmplen
1850 \ifcsundef{hypertarget}%
1851 {%
1852 \gdef@glstarget#1#2{#2}%
1853 }%
1854 {%
1855 \gdef@glstarget#1#2{%

```

```

1856 \settoheight{\gls@tmplen}{#2}%
1857 \raisebox{\gls@tmplen}{\hypertarget{#1}{}}#2%
1858 }%
1859 }

```

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

`\glsdisablehyper`

```

1860 \newcommand{\glsdisablehyper}{%
1861 \renewcommand*\@glslink[2]{##2}%
1862 \renewcommand*\@glsstarget[2]{##2}}

```

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

`\glsenablehyper`

```

1863 \newcommand{\glsenablehyper}{%
1864 \renewcommand*\@glslink[2]{\hyperlink{##1}{##2}}%
1865 \renewcommand*\@glsstarget[2]{%
1866 \settoheight{\gls@tmplen}{##2}%
1867 \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
1868 \newrobustcmd*\@gls{\@ifstar\@sgls\@gls}

```

Define the starred form:

`\@sgls`

```

1869 \newcommand*\@sgls[1][\@gls[hyper=false,#1]]

```

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

\@gls

```
1870 \newcommand*{\@gls}[2][\@gls@#1]{%
1871   \new@ifnextchar[\@gls@{#1}{#2}}{\@gls@{#1}{#2}[\@gls@#1]}%
1872 }
```

\@gls@ Read in the final optional argument:

```
1873 \def\@gls@#1#2[#3]{%
1874   \glsdoifexists{#2}%
1875   {%
1876     \edef\@glo@type{\glsentrytype{#2}}%
1877     \def\@gls@link@opts{#1}%
1878     \def\@gls@link@label{#2}%
1879     \ifglsused{#2}%
1880     {%
1881       \def\@glo@text{%
1882         \csname gls@\@glo@type @display\endcsname
1883         {\glsentrytext{#2}}{\glsentrydesc{#2}}{\glsentrysymbol{#2}}{#3}}%
1884       }%
1885     {%
1886       \def\@glo@text{%
1887         \csname gls@\@glo@type @displayfirst\endcsname
1888         {\glsentryfirst{#2}}{\glsentrydesc{#2}}{\glsentrysymbol{#2}}{#3}}%
1889     }%
1890   }
```

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.

```
1890   \ifglsused{#2}%
1891   {%
1892     \@gls@link[#1]{#2}{\@glo@text}%
1893   }%
1894   {%
1895     \gls@checkisacronymlist\@glo@type
1896     \ifthenelse
1897       {\(\boolean{glsisacronymlist}\AND \boolean{glsacrfootnote}\)}
1898       {\OR \NOT\boolean{gls hyperfirst}}
1899     {%
1900       \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
1901     }%
1902   }%
1903   {%
1904     \@gls@link[#1]{#2}{\@glo@text}%
1905   }%
1906 }
```

Indicate that this entry has now been used

```

1907 \ifKV@glslink@local
1908 \glsllocalunset{#2}%
1909 \else
1910 \glsunset{#2}%
1911 \fi
1912 }%
1913 }

```

`\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`

```

1914 \newrobustcmd*{\Gls}{\@ifstar\@sGls\@Gls}

```

Define the starred form:

```

1915 \newcommand*{\@sGls}[1] [] {\@Gls[hyper=false,#1]}

```

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

```

1916 \newcommand*{\@Gls}[2] [] {%
1917 \new@ifnextchar[{\@Gls@{#1}{#2}}{\@Gls@{#1}{#2} []}]%
1918 }

```

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

```

1919 \def\@Gls@#1#2[#3]{%
1920 \glsdoifexists{#2}%
1921 {%
1922 \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

1923 \def\@gls@link@opts{#1}%
1924 \def\@gls@link@label{#2}%
1925 \def\@gls@label{#2}%

```

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

```

1926 \ifglsused{#2}%
1927 {%
1928 \protected@edef\@glo@text{%
1929 \csname gls@\@glo@type @display\endcsname
1930 {\glsentrytext{#2}}{\glsentrydesc{#2}}%
1931 {\glsentrysymbol{#2}}{#3}}%
1932 }%
1933 {%
1934 \protected@edef\@glo@text{%
1935 \csname gls@\@glo@type @displayfirst\endcsname
1936 {\glsentryfirst{#2}}{\glsentrydesc{#2}}%
1937 {\glsentrysymbol{#2}}{#3}}%
1938 }%

```

Call `\@gls@link` If footnote package option has been used and the glossary type is `\acronymtype`, suppress hyperlink for first use. Likewise if the `hyper-first=false` package option is used.

```

1939   \ifglsused{#2}%
1940   {%
1941     \@gls@link[#1]{#2}{%
1942       \expandafter\makefirstuc\expandafter{\@glo@text}}%
1943   }%
1944   {%
1945     \gls@checkisacronymlist\@glo@type
1946     \ifthenelse
1947       {%
1948         \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
1949         \OR \NOT\boolean{glshyperfirst}}%
1950       {%
1951         \@gls@link[#1,hyper=false]{#2}{%
1952           \expandafter\makefirstuc\expandafter{\@glo@text}}%
1953       }%
1954       {%
1955         \@gls@link[#1]{#2}{%
1956           \expandafter\makefirstuc\expandafter{\@glo@text}}%
1957       }%
1958   }%
1959 }%
```

Indicate that this entry has now been used

```

1960   \ifKV@glslink@local
1961     \glslocalunset{#2}%
1962   \else
1963     \glsunset{#2}%
1964   \fi
1965 }%
1966 }
```

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

`\GLS`

```

1967 \newrobustcmd*{\GLS}{\@ifstar\@sGLS\@GLS}
```

Define the starred form:

```

1968 \newcommand*{\@sGLS}[1] [] {\@GLS[hyper=false,#1]}
```

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

```

1969 \newcommand*{\@GLS}[2] [] {%
1970   \new@ifnextchar[\@GLS@{#1}{#2}}{\@GLS@{#1}{#2} []}%
1971 }
```

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

```

1972 \def\@GLS@#1#2[#3]{%
1973   \glsdoifexists{#2}%
1974   {%
1975     \edef\@glo@type{\glsentrytype{#2}}%
      Save options in \@gls@link@opts and label in \@gls@link@label
1976     \def\@gls@link@opts{#1}%
1977     \def\@gls@link@label{#2}%
      Determine what the link text should be (this is stored in \@glo@text).
1978     \ifglsused{#2}%
1979     {%
1980       \def\@glo@text{%
1981         \csname gls@\@glo@type @display\endcsname
1982         {\glsentrytext{#2}}{\glsentrydesc{#2}}{\glsentrysymbol{#2}}{#3}%
1983       }%
1984     }%
1985     {%
1986       \def\@glo@text{%
1987         \csname gls@\@glo@type @displayfirst\endcsname
1988         {\glsentryfirst{#2}}{\glsentrydesc{#2}}{\glsentrysymbol{#2}}{#3}%
1989       }%
1990     }%
      Call \@gls@link If footnote package option has been used and the glossary
      type is \acronymtype, suppress hyperlink for first use. Likewise if the hyper-
      first=false package option is used.
1991     \ifglsused{#2}%
1992     {%
1993       \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
1994     }%
1995     {%
1996       \gls@checkisacronymlist\@glo@type
1997       \ifthenelse
1998       {%
1999         \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote}\)
2000         \OR \NOT\boolean{glshyperfirst}}{%
2001         \@gls@link[#1,hyper=false]{#2}{\MakeUppercase{\@glo@text}}%
2002       }%
2003     }%
2004     \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
2005     }%
2006   }%
      Indicate that this entry has now been used
2007   \ifKV@glslink@local
2008   \glslocalunset{#2}%
2009   \else
2010   \glsunset{#2}%
2011   \fi
2012 }%

```

2013 }

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

`\glspl`

2014 `\newrobustcmd*{\glspl}{\@ifstar\sglspl\glspl}`

Define the starred form:

2015 `\newcommand*{\sglspl}[1][\@glspl[hyper=false,#1]]`

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

2016 `\newcommand*{\@glspl}[2][\@%`

2017 `\new@ifnextchar[\@glspl@{#1}{#2}]{\@glspl@{#1}{#2}[]}%`

2018 }

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

2019 `\def\@glspl@#1#2[#3]{%`

2020 `\glsdoifexists{#2}%`

2021 `{%`

2022 `\edef\@glo@type{\glsentrytype{#2}}%`

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

2023 `\def\@gls@link@opts{#1}%`

2024 `\def\@gls@link@label{#2}%`

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

2025 `\ifglsused{#2}%`

2026 `{%`

2027 `\def\@glo@text{%`

2028 `\csname gls@\@glo@type @display\endcsname`

2029 `{\glsentryplural{#2}}{\glsentrydescplural{#2}}%`

2030 `{\glsentrysymbolplural{#2}}{#3}}%`

2031 `}%`

2032 `{%`

2033 `\def\@glo@text{%`

2034 `\csname gls@\@glo@type @displayfirst\endcsname`

2035 `{\glsentryfirstplural{#2}}{\glsentrydescplural{#2}}%`

2036 `{\glsentrysymbolplural{#2}}{#3}}%`

2037 `}%`

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.

2038 `\ifglsused{#2}%`

2039 `{%`

2040 `\@gls@link[#1]{#2}{\@glo@text}%`

2041 `}%`

2042 `{%`

2043 `\gls@checkisacronymlist\@glo@type`

2044 `\ifthenelse`

```

2045      {%
2046      \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote})\
2047      \OR \NOT\boolean{glshyperfirst}%
2048      }%
2049      {%
2050      \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
2051      }%
2052      {%
2053      \@gls@link[#1]{#2}{\@glo@text}%
2054      }%
2055      }%

```

Indicate that this entry has now been used

```

2056      \ifKV@glslink@local
2057      \glslocalunset{#2}%
2058      \else
2059      \glsunset{#2}%
2060      \fi
2061      }%
2062 }

```

`\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`

```

2063 \newrobustcmd*{\Glspl}{\@ifstar\@sGlspl\@Glspl}

```

Define the starred form:

```

2064 \newcommand*{\@sGlspl}[1][\@Glspl[hyper=false,#1]]

```

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

```

2065 \newcommand*{\@Glspl}[2][\@Glspl@{#1}{#2}]{\@Glspl@{#1}{#2}[]}
2066 \new@ifnextchar[\@Glspl@{#1}{#2}]{\@Glspl@{#1}{#2}[]}
2067 }

```

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

```

2068 \def\@Glspl@#1#2[#3]{%
2069   \glsdoifexists{#2}%
2070   {%
2071     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2072   \def\@gls@link@opts{#1}%
2073   \def\@gls@link@label{#2}%
2074   \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`.

```

2075 \ifglsused{#2}%
2076 {%
2077   \protected@edef\@glo@text{%
2078     \csname gls@\@glo@type @display\endcsname
2079     {\glsentryplural{#2}}{\glsentrydescplural{#2}}%
2080     {\glsentrysymbolplural{#2}}{#3}}%
2081   }%
2082 {%
2083   \protected@edef\@glo@text{%
2084     \csname gls@\@glo@type @displayfirst\endcsname
2085     {\glsentryfirstplural{#2}}{\glsentrydescplural{#2}}%
2086     {\glsentrysymbolplural{#2}}{#3}}%
2087   }%

```

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.

```

2088 \ifglsused{#2}%
2089 {%
2090   \@gls@link[#1]{#2}{%
2091     \expandafter\makefirstuc\expandafter{\@glo@text}}%
2092   }%
2093 {%
2094   \gls@checkisacronymlist\@glo@type
2095   \ifthenelse
2096     {%
2097       \(\boolean{glsisacronymlist}\AND \boolean{glsacrfootnote}\)
2098       \OR \NOT\boolean{glshyperfirst}}%
2099     {%
2100       \@gls@link[#1,hyper=false]{#2}{%
2101         \expandafter\makefirstuc\expandafter{\@glo@text}}%
2102       }%
2103     {%
2104       \@gls@link[#1]{#2}{%
2105         \expandafter\makefirstuc\expandafter{\@glo@text}}%
2106       }%
2107     }%
2108   }%

```

Indicate that this entry has now been used

```

2109 \ifKV@glslink@local
2110 \glslocalunset{#2}%
2111 \else
2112 \glsunset{#2}%
2113 \fi
2114 }%
2115 }

```

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

\GLSp1

```
2116 \newrobustcmd*{\GLSp1}{\@ifstar\@sGLSp1\@GLSp1}
```

Define the starred form:

```
2117 \newcommand*{\@sGLSp1}[1][]{\@GLSp1[hyper=false,#1]}
```

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

```
2118 \newcommand*{\@GLSp1}[2][]{%
```

```
2119   \new@ifnextchar[{\@GLSp1@{#1}{#2}}{\@GLSp1@{#1}{#2}[]}%
```

```
2120 }
```

\@GLSp1 Read in the final optional argument:

```
2121 \def\@GLSp1@#1#2[#3]{%
```

```
2122   \glsdoifexists{#2}%
```

```
2123   {%
```

```
2124     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2125     \def\@gls@link@opts{#1}%
```

```
2126     \def\@gls@link@label{#2}%
```

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

```
2127     \ifglsused{#2}%
```

```
2128     {%
```

```
2129       \def\@glo@text{%
```

```
2130         \csname gls@\@glo@type @display\endcsname
```

```
2131         {\glsentryplural{#2}}{\glsentrydescplural{#2}}%
```

```
2132         {\glsentrysymbolplural{#2}}{#3}%
```

```
2133       }%
```

```
2134     }%
```

```
2135     {%
```

```
2136       \def\@glo@text{%
```

```
2137         \csname gls@\@glo@type @displayfirst\endcsname
```

```
2138         {\glsentryfirstplural{#2}}{\glsentrydescplural{#2}}%
```

```
2139         {\glsentrysymbolplural{#2}}{#3}%
```

```
2140       }%
```

```
2141     }%
```

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.

```
2142     \ifglsused{#2}%
```

```
2143     {%
```

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

```
2145     }%
```

```
2146     {%
```

```
2147       \gls@checkisacronymlist\@glo@type
```

```
2148       \ifthenelse
```

```
2149       {%
```

```

2150      \(\boolean{@glsisacronymlist}\AND \boolean{glsacrfootnote})\
2151      \OR \NOT\boolean{glshyperfirst}}%
2152    }%
2153    {%
2154      \@gls@link[#1,hyper=false]{#2}{\MakeUppercase{\@glo@text}}}%
2155    }%
2156    {%
2157      \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}}%
2158    }%
2159  }%

```

Indicate that this entry has now been used

```

2160    \ifKV@glslink@local
2161      \glslocalunset{#2}%
2162    \else
2163      \glsunset{#2}%
2164    \fi
2165  }%
2166 }

```

`\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:

```

2167 \newrobustcmd*{\glsdisp}{\@ifstar\sglsdisp\@glsdisp}

```

Define the starred form:

`\@sgls`

```

2168 \newcommand*{\@sglsdisp}[1][\@glsdisp[hyper=false,#1]]{

```

Defined the un-starred form.

`\@glsdisp`

```

2169 \newcommand*{\@glsdisp}[3][\@glsdoifexists{#2}]{%
2170   \glsdoifexists{#2}{%
2171     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2172   \def\@gls@link@opts{#1}%
2173   \def\@gls@link@label{#2}%

```

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

```

2174   \ifglsused{#2}%
2175   {%
2176     \def\@glo@text{%
2177       \csname gls@\@glo@type @display\endcsname
2178       {#3}{\glsentrydesc{#2}}{\glsentrysymbol{#2}}{}}%
2179   }%

```

```

2180    {%
2181      \def\@glo@text{%
2182        \csname gls@\@glo@type @displayfirst\endcsname
2183        {#3}\@glsentrydesc{#2}}{\@glsentrysymbol{#2}}{}}%
2184    }%

```

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.

```

2185    \ifglsused{#2}%
2186    {%
2187      \@gls@link[#1]{#2}{\@glo@text}%
2188    }%
2189    {%
2190      \gls@checkisacronymlist\@glo@type
2191      \ifthenelse{(\boolean{@glsisacronymlist})\AND
2192        \boolean{glsacrfootnote}) \OR \NOT\boolean{glshyperfirst}}%
2193      {%
2194        \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
2195      }%
2196      {%
2197        \@gls@link[#1]{#2}{\@glo@text}%
2198      }%
2199    }%

```

Indicate that this entry has now been used

```

2200    \ifKV@glslink@local
2201      \glslocalunset{#2}%
2202    \else
2203      \glsunset{#2}%
2204    \fi
2205  }%
2206 }

```

`\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`

```

2207 \newrobustcmd*{\glstext}{\@ifstar\@sglstext\@glstext}

```

Define the starred form:

```

2208 \newcommand*{\@sglstext}[1][\@glstext[hyper=false,#1]]

```

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

```

2209 \newcommand*{\@glstext}[2][\@glstext@{#1}{#2}]{\@glstext@{#1}{#2}[]}
2210 \new@ifnextchar[\@glstext@{#1}{#2}]{\@glstext@{#1}{#2}[]}

```

Read in the final optional argument:

```

2211 \def\@glstext@#1#2[#3]{%
2212 \glsdoifexists{#2}{\edef\@glo@type{\@glsentrytype{#2}}%

```

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

```
2213 \protected@edef\@glo@text{\glentrytext{#2}}%
```

Call \@gls@link

```
2214 \@gls@link[#1]{#2}{\@glo@text#3}%
2215 }%
2216 }
```

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

\GLStext

```
2217 \newrobustcmd*{\GLStext}{\@ifstar\@sGLStext\@GLStext}
```

Define the starred form:

```
2218 \newcommand*{\@sGLStext}[1][]{\@GLStext[hyper=false,#1]}
```

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

```
2219 \newcommand*{\@GLStext}[2][]{%
2220 \new@ifnextchar[{\@GLStext@{#1}{#2}}{\@GLStext@{#1}{#2}[]}]}
```

Read in the final optional argument:

```
2221 \def\@GLStext@#1#2[#3]{%
2222 \glsdoifexists{#2}{\edef\@glo@type{\glentrytype{#2}}%
```

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

```
2223 \protected@edef\@glo@text{\glentrytext{#2}}%
```

Call \@gls@link

```
2224 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2225 }%
2226 }
```

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

\Glstext

```
2227 \newrobustcmd*{\Glstext}{\@ifstar\@sGlstext\@Glstext}
```

Define the starred form:

```
2228 \newcommand*{\@sGlstext}[1][]{\@Glstext[hyper=false,#1]}
```

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

```
2229 \newcommand*{\@Glstext}[2][]{%
2230 \new@ifnextchar[{\@Glstext@{#1}{#2}}{\@Glstext@{#1}{#2}[]}]}
```

Read in the final optional argument:

```
2231 \def\@Glstext@#1#2[#3]{%
2232 \glsdoifexists{#2}{\edef\@glo@type{\glentrytype{#2}}%
```

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

```
2233 \protected@edef\@glo@text{\glentrytext{#2}}%
```

Call \@gls@link

```

2234 \@gls@link[#1]{#2}{%
2235     \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2236 }%
2237 }

```

\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

```

2238 \newrobustcmd*{\glsfirst}{\@ifstar\sglsfirst\@glsfirst}

```

Define the starred form:

```

2239 \newcommand*{\@sglsfirst}[1][\@glsfirst[hyper=false,#1]]

```

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

```

2240 \newcommand*{\@glsfirst}[2][\@glsfirst@{#1}{#2}[]{}]
2241 \new@ifnextchar[\@glsfirst@{#1}{#2}]{\@glsfirst@{#1}{#2}[]{}]

```

Read in the final optional argument:

```

2242 \def\@glsfirst@#1#2[#3]{%
2243 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2244 \protected@edef\@glo@text{\glsentryfirst{#2}}%

```

Call \@gls@link

```

2245 \@gls@link[#1]{#2}{\@glo@text#3}%
2246 }%
2247 }

```

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

\Glsfirst

```

2248 \newrobustcmd*{\Glsfirst}{\@ifstar\sglsfirst\@Glsfirst}

```

Define the starred form:

```

2249 \newcommand*{\@sglsfirst}[1][\@Glsfirst[hyper=false,#1]]

```

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

```

2250 \newcommand*{\@Glsfirst}[2][\@Glsfirst@{#1}{#2}[]{}]
2251 \new@ifnextchar[\@Glsfirst@{#1}{#2}]{\@Glsfirst@{#1}{#2}[]{}]

```

Read in the final optional argument:

```

2252 \def\@Glsfirst@#1#2[#3]{%
2253 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2254 \protected@edef\@glo@text{\glsentryfirst{#2}}%

```

Call \@gls@link  
2255 \@gls@link[#1]{#2}{%  
2256 \expandafter\makefirstuc\expandafter{\@glo@text#3}%  
2257 }%  
2258 }

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

\GLSfirst

2259 \newrobustcmd\*{\GLSfirst}{\@ifstar\@sGLSfirst\@GLSfirst}

Define the starred form:

2260 \newcommand\*{\@sGLSfirst}[1][\@GLSfirst[hyper=false,#1]]

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

2261 \newcommand\*{\@GLSfirst}[2][\@GLSfirst@{#1}{#2}[]]

2262 \new@ifnextchar[\@GLSfirst@{#1}{#2}]{\@GLSfirst@{#1}{#2}[]}]

Read in the final optional argument:

2263 \def\@GLSfirst@#1#2[#3]{%

2264 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

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

2265 \protected@edef\@glo@text{\glsentryfirst{#2}}%

Call \@gls@link

2266 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%

2267 }%

2268 }

\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

2269 \newrobustcmd\*{\glsplural}{\@ifstar\@sglsplural\@glsplural}

Define the starred form:

2270 \newcommand\*{\@sglsplural}[1][\@glsplural[hyper=false,#1]]

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

2271 \newcommand\*{\@glsplural}[2][\@glsplural@{#1}{#2}[]]

2272 \new@ifnextchar[\@glsplural@{#1}{#2}]{\@glsplural@{#1}{#2}[]}]

Read in the final optional argument:

2273 \def\@glsplural@#1#2[#3]{%

2274 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

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

2275 \protected@edef\@glo@text{\glsentryplural{#2}}%

Call \@gls@link  
2276 \@gls@link[#1]{#2}{\@glo@text#3}%  
2277 }%  
2278 }

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

\Glsplural

2279 \newrobustcmd\*{\Glsplural}{\@ifstar\@sGlsplural\@Glsplural}

Define the starred form:

2280 \newcommand\*{\@sGlsplural}[1][\@Glsplural[hyper=false,#1]]

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

2281 \newcommand\*{\@Glsplural}[2][\@%

2282 \new@ifnextchar[\@Glsplural@{#1}{#2}]{\@Glsplural@{#1}{#2}[]}}

Read in the final optional argument:

2283 \def\@Glsplural@#1#2[#3]{%

2284 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

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

2285 \protected@edef\@glo@text{\glsentryplural{#2}}%

Call \@gls@link

2286 \@gls@link[#1]{#2}{%

2287 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%

2288 }%

2289 }

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

\GLSplural

2290 \newrobustcmd\*{\GLSplural}{\@ifstar\@sGLSplural\@GLSplural}

Define the starred form:

2291 \newcommand\*{\@sGLSplural}[1][\@GLSplural[hyper=false,#1]]

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

2292 \newcommand\*{\@GLSplural}[2][\@%

2293 \new@ifnextchar[\@GLSplural@{#1}{#2}]{\@GLSplural@{#1}{#2}[]}}

Read in the final optional argument:

2294 \def\@GLSplural@#1#2[#3]{%

2295 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

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

2296 \protected@edef\@glo@text{\glsentryplural{#2}}%

```

2297 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2298 }%
2299 }

```

\glsfirstplural

2316 \protected@edef\@glo@text{\glsentryfirstplural{#2}}%

Call \@gls@link

```

2317 \@gls@link[#1]{#2}{%
2318   \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2319 }%
2320 }

```

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

\GLSfirstplural

```

2321 \newrobustcmd*{\GLSfirstplural}{\@ifstar\@sGLSfirstplural\GLSfirstplural}

```

Define the starred form:

```

2322 \newcommand*{\@sGLSfirstplural}[1][\@GLSfirstplural[hyper=false,#1]]{

```

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

```

2323 \newcommand*{\@GLSfirstplural}[2][\@GLSfirstplural@#1]{#2}{\@GLSfirstplural@#1}{#2}[]}}
2324 \new@ifnextchar[\@GLSfirstplural@#1]{#2}{\@GLSfirstplural@#1}{#2}[]}}

```

Read in the final optional argument:

```

2325 \def\@GLSfirstplural@#1#2[#3]{%
2326 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

```

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

```

2327 \protected@edef\@glo@text{\glsentryfirstplural{#2}}%

```

Call \@gls@link

```

2328 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2329 }%
2330 }

```

\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

```

2331 \newrobustcmd*{\glsname}{\@ifstar\@sglsname\glsname}

```

Define the starred form:

```

2332 \newcommand*{\@sglsname}[1][\@glsname[hyper=false,#1]]{

```

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

```

2333 \newcommand*{\@glsname}[2][\@glsname@#1]{#2}{\@glsname@#1}{#2}[]}}
2334 \new@ifnextchar[\@glsname@#1]{#2}{\@glsname@#1}{#2}[]}}

```

Read in the final optional argument:

```

2335 \def\@glsname@#1#2[#3]{%
2336 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

```

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

```

2337 \protected@edef\@glo@text{\glsentryname{#2}}%

```

Call \@gls@link  
2338 \@gls@link[#1]{#2}{\@glo@text#3}%  
2339 }%  
2340 }

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

\Glsname

2341 \newrobustcmd\*{\Glsname}{\@ifstar\@sGlsname\@Glsname}

Define the starred form:

2342 \newcommand\*{\@sGlsname}[1][]{\@Glsname[hyper=false,#1]}

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

2343 \newcommand\*{\@Glsname}[2][]{%  
2344 \new@ifnextchar[{\@Glsname@{#1}{#2}}{\@Glsname@{#1}{#2}[]}]}

Read in the final optional argument:

2345 \def\@Glsname@#1#2[#3]{%  
2346 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

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

2347 \protected@edef\@glo@text{\glsentryname{#2}}%

Call \@gls@link

2348 \@gls@link[#1]{#2}{%  
2349 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%  
2350 }%  
2351 }

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

\GLSname

2352 \newrobustcmd\*{\GLSname}{\@ifstar\@sGLSname\@GLSname}

Define the starred form:

2353 \newcommand\*{\@sGLSname}[1][]{\@GLSname[hyper=false,#1]}

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

2354 \newcommand\*{\@GLSname}[2][]{%  
2355 \new@ifnextchar[{\@GLSname@{#1}{#2}}{\@GLSname@{#1}{#2}[]}]}

Read in the final optional argument:

2356 \def\@GLSname@#1#2[#3]{%  
2357 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

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

2358 \protected@edef\@glo@text{\glsentryname{#2}}%

Call \@gls@link

```
2359 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%  
2360 }%  
2361 }
```

\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

```
2362 \newrobustcmd*{\glsdesc}{\@ifstar\sglsdesc\@glsdesc}
```

Define the starred form:

```
2363 \newcommand*{\@sglsdesc}[1][\@glsdesc[hyper=false,#1]]
```

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

```
2364 \newcommand*{\@glsdesc}[2][\@%  
2365 \new@ifnextchar[\@glsdesc@{#1}{#2}]{\@glsdesc@{#1}{#2}[]}]
```

Read in the final optional argument:

```
2366 \def\@glsdesc@#1#2[#3]{%  
2367 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2368 \protected@edef\@glo@text{\glsentrydesc{#2}}%
```

Call \@gls@link

```
2369 \@gls@link[#1]{#2}{\@glo@text#3}%  
2370 }%  
2371 }
```

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

\Glsdesc

```
2372 \newrobustcmd*{\Glsdesc}{\@ifstar\sglsdesc\@Glsdesc}
```

Define the starred form:

```
2373 \newcommand*{\@sglsdesc}[1][\@Glsdesc[hyper=false,#1]]
```

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

```
2374 \newcommand*{\@Glsdesc}[2][\@%  
2375 \new@ifnextchar[\@Glsdesc@{#1}{#2}]{\@Glsdesc@{#1}{#2}[]}]
```

Read in the final optional argument:

```
2376 \def\@Glsdesc@#1#2[#3]{%  
2377 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2378 \protected@edef\@glo@text{\glsentrydesc{#2}}%
```

Call \@gls@link  
2379 \@gls@link[#1]{#2}{%  
2380 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%  
2381 }%  
2382 }  
\GLSdesc behaves like \glsdesc except that the link text is converted to uppercase.

\GLSdesc

2383 \newrobustcmd\*{\GLSdesc}{\@ifstar\@sGLSdesc\@GLSdesc}  
Define the starred form:  
2384 \newcommand\*{\@sGLSdesc}[1] [] {\@GLSdesc[hyper=false,#1]}  
Defined the un-starred form. Need to determine if there is a final optional argument  
2385 \newcommand\*{\@GLSdesc}[2] [] {%  
2386 \new@ifnextchar [\@GLSdesc@{#1}{#2}]{\@GLSdesc@{#1}{#2} []}}  
Read in the final optional argument:  
2387 \def\@GLSdesc@#1#2[#3] {%  
2388 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%  
Determine what the link text should be (this is stored in \@glo@text)  
2389 \protected@edef\@glo@text{\glsentrydesc{#2}}%  
Call \@gls@link  
2390 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%  
2391 }%  
2392 }  
\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

2393 \newrobustcmd\*{\glsdescplural}{\@ifstar\@sglsdescplural\@glsdescplural}  
Define the starred form:  
2394 \newcommand\*{\@sglsdescplural}[1] [] {\@glsdescplural[hyper=false,#1]}  
Defined the un-starred form. Need to determine if there is a final optional argument  
2395 \newcommand\*{\@glsdescplural}[2] [] {%  
2396 \new@ifnextchar [\@glsdescplural@{#1}{#2}]{\@glsdescplural@{#1}{#2} []}}  
Read in the final optional argument:  
2397 \def\@glsdescplural@#1#2[#3] {%  
2398 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%  
Determine what the link text should be (this is stored in \@glo@text)  
2399 \protected@edef\@glo@text{\glsentrydescplural{#2}}%

Call \@gls@link  
2400 \@gls@link[#1]{#2}{\@glo@text#3}%  
2401 }%  
2402 }  
\Glsdescplural behaves like \glsdescplural except that the first letter is converted to uppercase.

\Glsdescplural

2403 \newrobustcmd\*{\Glsdescplural}{\@ifstar\@sGlsdescplural\@Glsdescplural}

Define the starred form:

2404 \newcommand\*{\@sGlsdescplural}[1][\@Glsdescplural[hyper=false,#1]]

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

2405 \newcommand\*{\@Glsdescplural}[2][\@Glsdescplural@{#1}{#2}[]]

2406 \new@ifnextchar[\@Glsdescplural@{#1}{#2}]{\@Glsdescplural@{#1}{#2}[]}]

Read in the final optional argument:

2407 \def\@Glsdescplural@#1#2[#3]{%

2408 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

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

2409 \protected@edef\@glo@text{\glsentrydescplural{#2}}%

Call \@gls@link

2410 \@gls@link[#1]{#2}{%

2411 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%

2412 }%

2413 }

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

\GLSdescplural

2414 \newrobustcmd\*{\GLSdescplural}{\@ifstar\@sGLSdescplural\@GLSdescplural}

Define the starred form:

2415 \newcommand\*{\@sGLSdescplural}[1][\@GLSdescplural[hyper=false,#1]]

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

2416 \newcommand\*{\@GLSdescplural}[2][\@GLSdescplural@{#1}{#2}[]]

2417 \new@ifnextchar[\@GLSdescplural@{#1}{#2}]{\@GLSdescplural@{#1}{#2}[]}]

Read in the final optional argument:

2418 \def\@GLSdescplural@#1#2[#3]{%

2419 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

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

2420 \protected@edef\@glo@text{\glsentrydescplural{#2}}%

Call \@gls@link

```
2421 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%  
2422 }%  
2423 }
```

\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

```
2424 \newrobustcmd*{\glsymbol}{\@ifstar\sglsymbol\@glsymbol}
```

Define the starred form:

```
2425 \newcommand*{\sglsymbol}[1][\@glsymbol[hyper=false,#1]]
```

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

```
2426 \newcommand*{\@glsymbol}[2][\@glsymbol@{#1}{#2}[]]{%  
2427 \new@ifnextchar[\@glsymbol@{#1}{#2}]{\@glsymbol@{#1}{#2}[]}}
```

Read in the final optional argument:

```
2428 \def\@glsymbol@#1#2[#3]{%  
2429 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2430 \protected@edef\@glo@text{\glsentrysymbol{#2}}%
```

Call \@gls@link

```
2431 \@gls@link[#1]{#2}{\@glo@text#3}%  
2432 }%  
2433 }
```

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

\Glsymbol

```
2434 \newrobustcmd*{\Glsymbol}{\@ifstar\sglsymbol\@Glsymbol}
```

Define the starred form:

```
2435 \newcommand*{\sglsymbol}[1][\@Glsymbol[hyper=false,#1]]
```

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

```
2436 \newcommand*{\@Glsymbol}[2][\@Glsymbol@{#1}{#2}[]]{%  
2437 \new@ifnextchar[\@Glsymbol@{#1}{#2}]{\@Glsymbol@{#1}{#2}[]}}
```

Read in the final optional argument:

```
2438 \def\@Glsymbol@#1#2[#3]{%  
2439 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2440 \protected@edef\@glo@text{\glsentrysymbol{#2}}%
```

Call \@gls@link

```

2441 \@gls@link[#1]{#2}{%
2442   \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2443 }%
2444 }

```

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

\GLSsymbol

```

2445 \newrobustcmd*{\GLSsymbol}{\@ifstar\@sGLSsymbol\@GLSsymbol}

```

Define the starred form:

```

2446 \newcommand*{\@sGLSsymbol}[1][\@GLSsymbol[hyper=false,#1]]{

```

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

```

2447 \newcommand*{\@GLSsymbol}[2][\@GLSsymbol@{#1}{#2}[]]{%
2448 \new@ifnextchar[\@GLSsymbol@{#1}{#2}]{\@GLSsymbol@{#1}{#2}[]}}

```

Read in the final optional argument:

```

2449 \def\@GLSsymbol@#1#2[#3]{%
2450 \glsoifexists{#2}{\edef\@glo@type{\glstrytype{#2}}%

```

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

```

2451 \protected@edef\@glo@text{\glstrytype{#2}}%

```

Call \@gls@link

```

2452 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2453 }%
2454 }

```

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

\glssymbolplural

```

2455 \newrobustcmd*{\glssymbolplural}{\@ifstar\@sglssymbolplural\@glssymbolplural}

```

Define the starred form:

```

2456 \newcommand*{\@sglssymbolplural}[1][\@glssymbolplural[hyper=false,#1]]{

```

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

```

2457 \newcommand*{\@glssymbolplural}[2][\@glssymbolplural@{#1}{#2}[]]{%
2458 \new@ifnextchar[\@glssymbolplural@{#1}{#2}]{\@glssymbolplural@{#1}{#2}[]}}

```

Read in the final optional argument:

```

2459 \def\@glssymbolplural@#1#2[#3]{%
2460 \glsoifexists{#2}{\edef\@glo@type{\glstrytype{#2}}%

```

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

```

2461 \protected@edef\@glo@text{\glstrytype{#2}}%

```

Call \@gls@link

```
2462 \@gls@link[#1]{#2}{\@glo@text#3}%  
2463 }%  
2464 }
```

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

\Glsymbolplural

```
2465 \newrobustcmd*{\Glsymbolplural}{\@ifstar\@sGlsymbolplural\@Glsymbolplural}
```

Define the starred form:

```
2466 \newcommand*{\@sGlsymbolplural}[1][\@Glsymbolplural[hyper=false,#1]]{
```

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

```
2467 \newcommand*{\@Glsymbolplural}[2][\@%  
2468 \new@ifnextchar[\@Glsymbolplural@{#1}{#2}]{\@Glsymbolplural@{#1}{#2}[]}}
```

Read in the final optional argument:

```
2469 \def\@Glsymbolplural@#1#2[#3]{%  
2470 \glsdoifexists{#2}{\edef\@glo@type{\glstrytype{#2}}}%
```

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

```
2471 \protected@edef\@glo@text{\glstrysymbolplural{#2}}%
```

Call \@gls@link

```
2472 \@gls@link[#1]{#2}{%  
2473 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%  
2474 }%  
2475 }
```

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

\GLSsymbolplural

```
2476 \newrobustcmd*{\GLSsymbolplural}{\@ifstar\@sGLSsymbolplural\@GLSsymbolplural}
```

Define the starred form:

```
2477 \newcommand*{\@sGLSsymbolplural}[1][\@GLSsymbolplural[hyper=false,#1]]{
```

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

```
2478 \newcommand*{\@GLSsymbolplural}[2][\@%  
2479 \new@ifnextchar[\@GLSsymbolplural@{#1}{#2}]{\@GLSsymbolplural@{#1}{#2}[]}}
```

Read in the final optional argument:

```
2480 \def\@GLSsymbolplural@#1#2[#3]{%  
2481 \glsdoifexists{#2}{\edef\@glo@type{\glstrytype{#2}}}%
```

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

```
2482 \protected@edef\@glo@text{\glstrysymbolplural{#2}}%
```

Call \@gls@link

```

2483 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2484 }%
2485 }

```

\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

```

2486 \newrobustcmd*{\glsuseri}{\@ifstar\sglsuseri\@glsuseri}

```

Define the starred form:

```

2487 \newcommand*{\@sglsuseri}[1][\@glsuseri[hyper=false,#1]]

```

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

```

2488 \newcommand*{\@glsuseri}[2][\@%
2489 \new@ifnextchar[\@glsuseri@{#1}{#2}]{\@glsuseri@{#1}{#2}[]}]

```

Read in the final optional argument:

```

2490 \def\@glsuseri@#1#2[#3]{%
2491 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2492 \protected@edef\@glo@text{\glsentryuseri{#2}}%

```

Call \@gls@link

```

2493 \@gls@link[#1]{#2}{\@glo@text#3}%
2494 }%
2495 }

```

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

\Glsuseri

```

2496 \newrobustcmd*{\Glsuseri}{\@ifstar\sglsuseri\@Glsuseri}

```

Define the starred form:

```

2497 \newcommand*{\@sglsuseri}[1][\@Glsuseri[hyper=false,#1]]

```

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

```

2498 \newcommand*{\@Glsuseri}[2][\@%
2499 \new@ifnextchar[\@Glsuseri@{#1}{#2}]{\@Glsuseri@{#1}{#2}[]}]

```

Read in the final optional argument:

```

2500 \def\@Glsuseri@#1#2[#3]{%
2501 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2502 \protected@edef\@glo@text{\glsentryuseri{#2}}%

```

Call \@gls@link

```

2503 \@gls@link[#1]{#2}{%
2504   \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2505 }%
2506 }

```

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

\GLSuseri

```

2507 \newrobustcmd*{\GLSuseri}{\@ifstar\@sGLSuseri\@GLSuseri}

```

Define the starred form:

```

2508 \newcommand*{\@sGLSuseri}[1][\@GLSuseri[hyper=false,#1]]{

```

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

```

2509 \newcommand*{\@GLSuseri}[2][\@GLSuseri@{#1}{#2}]{\@GLSuseri@{#1}{#2}[]}
2510 \new@ifnextchar[\@GLSuseri@{#1}{#2}]{\@GLSuseri@{#1}{#2}[]}

```

Read in the final optional argument:

```

2511 \def\@GLSuseri@#1#2[#3]{%
2512 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2513 \protected@edef\@glo@text{\glsentryuseri{#2}}%

```

Call \@gls@link

```

2514 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2515 }%
2516 }

```

\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

```

2517 \newrobustcmd*{\glsuserii}{\@ifstar\@sglsuserii\@glsuserii}

```

Define the starred form:

```

2518 \newcommand*{\@sglsuserii}[1][\@glsuserii[hyper=false,#1]]{

```

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

```

2519 \newcommand*{\@glsuserii}[2][\@glsuserii@{#1}{#2}]{\@glsuserii@{#1}{#2}[]}
2520 \new@ifnextchar[\@glsuserii@{#1}{#2}]{\@glsuserii@{#1}{#2}[]}

```

Read in the final optional argument:

```

2521 \def\@glsuserii@#1#2[#3]{%
2522 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2523 \protected@edef\@glo@text{\glsentryuserii{#2}}%

```

Call \@gls@link  
2524 \@gls@link[#1]{#2}{\@glo@text#3}%  
2525 }%  
2526 }

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

\Glsuserii

2527 \newrobustcmd\*{\Glsuserii}{\@ifstar\@sGlsuserii\@Glsuserii}

Define the starred form:

2528 \newcommand\*{\@sGlsuserii}[1][\@Glsuserii[hyper=false,#1]]

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

2529 \newcommand\*{\@Glsuserii}[2][\@%

2530 \new@ifnextchar[\@Glsuserii@{#1}{#2}]{\@Glsuserii@{#1}{#2}[]}}

Read in the final optional argument:

2531 \def\@Glsuserii@#1#2[#3]{%

2532 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

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

2533 \protected@edef\@glo@text{\glsentryuserii{#2}}%

Call \@gls@link

2534 \@gls@link[#1]{#2}{%

2535 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%

2536 }%

2537 }

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

\GLSuserii

2538 \newrobustcmd\*{\GLSuserii}{\@ifstar\@sGLSuserii\@GLSuserii}

Define the starred form:

2539 \newcommand\*{\@sGLSuserii}[1][\@GLSuserii[hyper=false,#1]]

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

2540 \newcommand\*{\@GLSuserii}[2][\@%

2541 \new@ifnextchar[\@GLSuserii@{#1}{#2}]{\@GLSuserii@{#1}{#2}[]}}

Read in the final optional argument:

2542 \def\@GLSuserii@#1#2[#3]{%

2543 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

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

2544 \protected@edef\@glo@text{\glsentryuserii{#2}}%

Call \@gls@link  
2545 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%  
2546 }%  
2547 }

\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

2548 \newrobustcmd\*{\glsuseriii}{\@ifstar\@sglsuseriii\@glsuseriii}

Define the starred form:

2549 \newcommand\*{\@sglsuseriii}[1][\@glsuseriii[hyper=false,#1]]

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

2550 \newcommand\*{\@glsuseriii}[2][\@%  
2551 \new@ifnextchar[\@glsuseriii@{#1}{#2}]{\@glsuseriii@{#1}{#2}[]}]

Read in the final optional argument:

2552 \def\@glsuseriii@#1#2[#3]{%  
2553 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

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

2554 \protected@edef\@glo@text{\glsentryuseriii{#2}}%

Call \@gls@link

2555 \@gls@link[#1]{#2}{\@glo@text#3}}%  
2556 }%  
2557 }

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

\Glsuseriii

2558 \newrobustcmd\*{\Glsuseriii}{\@ifstar\@sGlsuseriii\@Glsuseriii}

Define the starred form:

2559 \newcommand\*{\@sGlsuseriii}[1][\@Glsuseriii[hyper=false,#1]]

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

2560 \newcommand\*{\@Glsuseriii}[2][\@%  
2561 \new@ifnextchar[\@Glsuseriii@{#1}{#2}]{\@Glsuseriii@{#1}{#2}[]}]

Read in the final optional argument:

2562 \def\@Glsuseriii@#1#2[#3]{%  
2563 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}}%

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

2564 \protected@edef\@glo@text{\glsentryuseriii{#2}}%

Call \@gls@link

```

2565 \@gls@link[#1]{#2}{%
2566   \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2567 }%
2568 }

```

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

\GLSuseriii

```

2569 \newrobustcmd*{\GLSuseriii}{\@ifstar\@sGLSuseriii\@GLSuseriii}

```

Define the starred form:

```

2570 \newcommand*{\@sGLSuseriii}[1] [] {\@GLSuseriii[hyper=false,#1]}

```

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

```

2571 \newcommand*{\@GLSuseriii}[2] [] {%
2572   \new@ifnextchar [\@GLSuseriii@{#1}{#2}]{\@GLSuseriii@{#1}{#2} []}}

```

Read in the final optional argument:

```

2573 \def\@GLSuseriii@#1#2[#3] {%
2574   \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2575   \protected@edef\@glo@text{\glsentryuseriii{#2}}%

```

Call \@gls@link

```

2576 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2577 }%
2578 }

```

\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

```

2579 \newrobustcmd*{\glsuseriv}{\@ifstar\@sglsuseriv\@glsuseriv}

```

Define the starred form:

```

2580 \newcommand*{\@sglsuseriv}[1] [] {\@glsuseriv[hyper=false,#1]}

```

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

```

2581 \newcommand*{\@glsuseriv}[2] [] {%
2582   \new@ifnextchar [\@glsuseriv@{#1}{#2}]{\@glsuseriv@{#1}{#2} []}}

```

Read in the final optional argument:

```

2583 \def\@glsuseriv@#1#2[#3] {%
2584   \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2585   \protected@edef\@glo@text{\glsentryuseriv{#2}}%

```

Call \@gls@link  
2586 \@gls@link[#1]{#2}{\@glo@text#3}%  
2587 }%  
2588 }

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

\Glsuseriv

2589 \newrobustcmd\*{\Glsuseriv}{\@ifstar\@sGlsuseriv\@Glsuseriv}

Define the starred form:

2590 \newcommand\*{\@sGlsuseriv}[1][]{\@Glsuseriv[hyper=false,#1]}

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

2591 \newcommand\*{\@Glsuseriv}[2][]{%

2592 \new@ifnextchar[\@Glsuseriv@{#1}{#2}]{\@Glsuseriv@{#1}{#2}[]}}

Read in the final optional argument:

2593 \def\@Glsuseriv@#1#2[#3]{%

2594 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

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

2595 \protected@edef\@glo@text{\glsentryuseriv{#2}}%

Call \@gls@link

2596 \@gls@link[#1]{#2}{%

2597 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%

2598 }%

2599 }

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

\GLSuseriv

2600 \newrobustcmd\*{\GLSuseriv}{\@ifstar\@sGLSuseriv\@GLSuseriv}

Define the starred form:

2601 \newcommand\*{\@sGLSuseriv}[1][]{\@GLSuseriv[hyper=false,#1]}

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

2602 \newcommand\*{\@GLSuseriv}[2][]{%

2603 \new@ifnextchar[\@GLSuseriv@{#1}{#2}]{\@GLSuseriv@{#1}{#2}[]}}

Read in the final optional argument:

2604 \def\@GLSuseriv@#1#2[#3]{%

2605 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

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

2606 \protected@edef\@glo@text{\glsentryuseriv{#2}}%

Call \@gls@link

```
2607 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%  
2608 }%  
2609 }
```

\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

```
2610 \newrobustcmd*{\glsuserv}{\@ifstar\sglsuserv\@glsuserv}
```

Define the starred form:

```
2611 \newcommand*{\@sglsuserv}[1][\@glsuserv[hyper=false,#1]]
```

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

```
2612 \newcommand*{\@glsuserv}[2][\@%  
2613 \new@ifnextchar[\@glsuserv@{#1}{#2}]{\@glsuserv@{#1}{#2}[]}]
```

Read in the final optional argument:

```
2614 \def\@glsuserv@#1#2[#3]{%  
2615 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2616 \protected@edef\@glo@text{\glsentryuser{#2}}%
```

Call \@gls@link

```
2617 \@gls@link[#1]{#2}{\@glo@text#3}%  
2618 }%  
2619 }
```

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

\Glsuserv

```
2620 \newrobustcmd*{\Glsuserv}{\@ifstar\sglsuserv\@Glsuserv}
```

Define the starred form:

```
2621 \newcommand*{\@sglsuserv}[1][\@Glsuserv[hyper=false,#1]]
```

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

```
2622 \newcommand*{\@Glsuserv}[2][\@%  
2623 \new@ifnextchar[\@Glsuserv@{#1}{#2}]{\@Glsuserv@{#1}{#2}[]}]
```

Read in the final optional argument:

```
2624 \def\@Glsuserv@#1#2[#3]{%  
2625 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2626 \protected@edef\@glo@text{\glsentryuser{#2}}%
```

Call \@gls@link

```
2627 \@gls@link[#1]{#2}{%
2628   \expandafter\makefirstuc\expandafter{\@glo@text}#3}%
2629 }%
2630 }
```

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

\GLSuserv

```
2631 \newrobustcmd*{\GLSuserv}{\@ifstar\@sGLSuserv\@GLSuserv}
```

Define the starred form:

```
2632 \newcommand*{\@sGLSuserv}[1][\@GLSuserv[hyper=false,#1]]
```

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

```
2633 \newcommand*{\@GLSuserv}[2][\@GLSuserv@{#1}{#2}]{\@GLSuserv@{#1}{#2}[]}
```

Read in the final optional argument:

```
2635 \def\@GLSuserv@#1#2[#3]{%
2636 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2637 \protected@edef\@glo@text{\glsentryuserv{#2}}%
```

Call \@gls@link

```
2638 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2639 }%
2640 }
```

\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

```
2641 \newrobustcmd*{\glsuservi}{\@ifstar\@sglsuservi\@glsuservi}
```

Define the starred form:

```
2642 \newcommand*{\@sglsuservi}[1][\@glsuservi[hyper=false,#1]]
```

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

```
2643 \newcommand*{\@glsuservi}[2][\@glsuservi@{#1}{#2}]{\@glsuservi@{#1}{#2}[]}
```

Read in the final optional argument:

```
2645 \def\@glsuservi@#1#2[#3]{%
2646 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2647 \protected@edef\@glo@text{\glsentryuservi{#2}}%
```

Call \@gls@link  
2648 \@gls@link[#1]{#2}{\@glo@text#3}%  
2649 }%  
2650 }

\Glsuservi behaves like \glsuservi except that the first letter is converted to uppercase.

\Glsuservi

2651 \newrobustcmd\*{\Glsuservi}{\@ifstar\@sGlsuservi\@Glsuservi}

Define the starred form:

2652 \newcommand\*{\@sGlsuservi}[1][\@Glsuservi[hyper=false,#1]]

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

2653 \newcommand\*{\@Glsuservi}[2][\@%

2654 \new@ifnextchar[\@Glsuservi@{#1}{#2}]{\@Glsuservi@{#1}{#2}[]}}

Read in the final optional argument:

2655 \def\@Glsuservi@#1#2[#3]{%

2656 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

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

2657 \protected@edef\@glo@text{\glsentryuservi{#2}}%

Call \@gls@link

2658 \@gls@link[#1]{#2}{%

2659 \expandafter\makefirstuc\expandafter{\@glo@text}#3}%

2660 }%

2661 }

\GLSuservi behaves like \glsuservi except that the link text is converted to uppercase.

\GLSuservi

2662 \newrobustcmd\*{\GLSuservi}{\@ifstar\@sGLSuservi\@GLSuservi}

Define the starred form:

2663 \newcommand\*{\@sGLSuservi}[1][\@GLSuservi[hyper=false,#1]]

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

2664 \newcommand\*{\@GLSuservi}[2][\@%

2665 \new@ifnextchar[\@GLSuservi@{#1}{#2}]{\@GLSuservi@{#1}{#2}[]}}

Read in the final optional argument:

2666 \def\@GLSuservi@#1#2[#3]{%

2667 \glsdoifexists{#2}{\edef\@glo@type{\glsentrytype{#2}}%

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

2668 \protected@edef\@glo@text{\glsentryuservi{#2}}%

Call \@gls@link

```
2669 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%  
2670 }%  
2671 }
```

Now deal with acronym related keys. First the short form:

\acrshort

```
2672 \newrobustcmd*{\acrshort}{\@ifstar\s@acrshort\ns@acrshort}
```

Define the starred form:

```
2673 \newcommand*{\s@acrshort}[2][]{%  
2674   \new@ifnextchar[{\@acrshort{hyper=false,#1}{#2}}%  
2675     {\@acrshort{hyper=false,#1}{#2}[]}%  
2676 }
```

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

```
2677 \newcommand*{\ns@acrshort}[2][]{%  
2678   \new@ifnextchar[{\@acrshort{#1}{#2}}{\@acrshort{#1}{#2}[]}%  
2679 }
```

Read in the final optional argument:

```
2680 \def\@acrshort#1#2[#3]{%  
2681   \glsdoifexists{#2}%  
2682   {%  
2683     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2684     \protected@edef\@glo@text{\glsentryshort{#2}}%
```

Call \@gls@link

```
2685     \@gls@link[#1]{#2}{\acronymfont{\@glo@text}#3}%  
2686   }%  
2687 }
```

\Acrshort

```
2688 \newrobustcmd*{\Acrshort}{\@ifstar\s@Acrshort\ns@Acrshort}
```

Define the starred form:

```
2689 \newcommand*{\s@Acrshort}[2][]{%  
2690   \new@ifnextchar[{\@Acrshort{hyper=false,#1}{#2}}%  
2691     {\@Acrshort{hyper=false,#1}{#2}[]}%  
2692 }
```

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

```
2693 \newcommand*{\ns@Acrshort}[2][]{%  
2694   \new@ifnextchar[{\@Acrshort{#1}{#2}}{\@Acrshort{#1}{#2}[]}%  
2695 }
```

Read in the final optional argument:

```
2696 \def\@Acrshort#1#2[#3]{%
2697   \glsdoifexists{#2}%
2698   {%
2699     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2700   \protected@edef\@glo@text{\glsentryshort{#2}}%
```

Call \@gls@link

```
2701   \@gls@link[#1]{#2}%
2702   {%
2703     \acronymfont{\expandafter\makefirstuc\expandafter{\@glo@text}}#3%
2704   }%
2705 }%
2706 }
```

\ACRshort

```
2707 \newrobustcmd*{\ACRshort}{\@ifstar\s@ACRshort\ns@ACRshort}
```

Define the starred form:

```
2708 \newcommand*{\s@ACRshort}[2][{}]{%
2709   \new@ifnextchar[{\@ACRshort{hyper=false,#1}{#2}}%
2710   {\@ACRshort{hyper=false,#1}{#2}[]}%
2711 }
```

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

```
2712 \newcommand*{\ns@ACRshort}[2][{}]{%
2713   \new@ifnextchar[{\@ACRshort{#1}{#2}}{\@ACRshort{#1}{#2}[]}%
2714 }
```

Read in the final optional argument:

```
2715 \def\@ACRshort#1#2[#3]{%
2716   \glsdoifexists{#2}%
2717   {%
2718     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2719   \protected@edef\@glo@text{\glsentryshort{#2}}%
```

Call \@gls@link

```
2720   \@gls@link[#1]{#2}{\acronymfont{\MakeUppercase{\@glo@text#3}}}%
2721   }%
2722 }
```

Short plural:

\acrshortpl

```
2723 \newrobustcmd*{\acrshortpl}{\@ifstar\s@acrshortpl\ns@acrshortpl}
```

Define the starred form:

```
2724 \newcommand*{\s@acrshortpl}[2] [] {%
2725   \new@ifnextchar[{\@acrshortpl{hyper=false,#1}{#2}}%
2726     {\@acrshortpl{hyper=false,#1}{#2} []}%
2727 }
```

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

```
2728 \newcommand*{\ns@acrshortpl}[2] [] {%
2729   \new@ifnextchar[{\@acrshortpl{#1}{#2}}{\@acrshortpl{#1}{#2} []}%
2730 }
```

Read in the final optional argument:

```
2731 \def\@acrshortpl#1#2[#3] {%
2732   \glsdoifexists{#2}%
2733   {%
2734     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2735   \protected@edef\@glo@text{\glsentryshortpl{#2}}%
```

Call \@gls@link

```
2736   \@gls@link[#1]{#2}{\acronymfont{\@glo@text}#3}%
2737 }%
2738 }
```

\Acrshortpl

```
2739 \newrobustcmd*{\Acrshortpl}{\@ifstar\s@Acrshortpl\ns@Acrshortpl}
```

Define the starred form:

```
2740 \newcommand*{\s@Acrshortpl}[2] [] {%
2741   \new@ifnextchar[{\@Acrshortpl{hyper=false,#1}{#2}}%
2742     {\@Acrshortpl{hyper=false,#1}{#2} []}%
2743 }
```

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

```
2744 \newcommand*{\ns@Acrshortpl}[2] [] {%
2745   \new@ifnextchar[{\@Acrshortpl{#1}{#2}}{\@Acrshortpl{#1}{#2} []}%
2746 }
```

Read in the final optional argument:

```
2747 \def\@Acrshortpl#1#2[#3] {%
2748   \glsdoifexists{#2}%
2749   {%
2750     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2751   \protected@edef\@glo@text{\glsentryshortpl{#2}}%
```

Call \@gls@link

```

2752   \@gls@link[#1]{#2}%
2753   {%
2754     \acronymfont{\expandafter\makefirstuc\expandafter{\@glo@text}}#3%
2755   }%
2756   }%
2757 }
```

\ACRshortpl

```
2758 \newrobustcmd*{\ACRshortpl}{\@ifstar\s@ACRshortpl\ns@ACRshortpl}
```

Define the starred form:

```

2759 \newcommand*{\s@ACRshortpl}[2] [] {%
2760   \new@ifnextchar[{\@ACRshortpl{hyper=false,#1}{#2}}%
2761     {\@ACRshortpl{hyper=false,#1}{#2} []}%
2762 }
```

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

```

2763 \newcommand*{\ns@ACRshortpl}[2] [] {%
2764   \new@ifnextchar[{\@ACRshortpl{#1}{#2}}{\@ACRshortpl{#1}{#2} []}%
2765 }
```

Read in the final optional argument:

```

2766 \def\@ACRshortpl#1#2[#3] {%
2767   \glsdoifexists{#2}%
2768   {%
2769     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2770   \protected@edef\@glo@text{\glsentryshortpl{#2}}%
```

Call \@gls@link

```

2771   \@gls@link[#1]{#2}{\acronymfont{\MakeUppercase{\@glo@text#3}}}%
2772   }%
2773 }
```

\acrlong

```
2774 \newrobustcmd*{\acrlong}{\@ifstar\s@acrlong\ns@acrlong}
```

Define the starred form:

```

2775 \newcommand*{\s@acrlong}[2] [] {%
2776   \new@ifnextchar[{\@acrlong{hyper=false,#1}{#2}}%
2777     {\@acrlong{hyper=false,#1}{#2} []}%
2778 }
```

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

```

2779 \newcommand*{\ns@acrlong}[2] [] {%
2780   \new@ifnextchar[{\@acrlong{#1}{#2}}{\@acrlong{#1}{#2} []}%
2781 }
```

Read in the final optional argument:

```
2782 \def\@acrlong#1#2[#3]{%
2783   \glsdoifexists{#2}%
2784   {%
2785     \edef\@glo@type{\glstrytype{#2}}%
2786     \protected@edef\@glo@text{\glstrylong{#2}}%
2787     \Call\@gls@link
2788   }%
2789 }
```

\Acrlong

```
2790 \newrobustcmd*{\Acrlong}{\@ifstar\s@Acrlong\@ns@Acrlong}
```

Define the starred form:

```
2791 \newcommand*{\s@Acrlong}[2][ ]{%
2792   \new@ifnextchar[{\@Acrlong{hyper=false,#1}{#2}}%
2793   {\@Acrlong{hyper=false,#1}{#2} [ ]}%
2794 }
```

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

```
2795 \newcommand*{\ns@Acrlong}[2][ ]{%
2796   \new@ifnextchar[{\@Acrlong{#1}{#2}}{\@Acrlong{#1}{#2} [ ]}%
2797 }
```

Read in the final optional argument:

```
2798 \def\@Acrlong#1#2[#3]{%
2799   \glsdoifexists{#2}%
2800   {%
2801     \edef\@glo@type{\glstrytype{#2}}%
2802     \protected@edef\@glo@text{\glstrylong{#2}}%
2803     \Call\@gls@link
2804     {%
2805       \expandafter\makefirstuc\expandafter{\@glo@text}#3%
2806     }%
2807   }%
2808 }
```

\ACRlong

```
2809 \newrobustcmd*{\ACRlong}{\@ifstar\s@ACRlong\@ns@ACRlong}
```

Define the starred form:

```
2810 \newcommand*{\s@ACRlong}[2] [] {%
2811   \new@ifnextchar[{\@ACRlong{hyper=false,#1}{#2}}%
2812     {\@ACRlong{hyper=false,#1}{#2} []}%
2813 }
```

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

```
2814 \newcommand*{\ns@ACRlong}[2] [] {%
2815   \new@ifnextchar[{\@ACRlong{#1}{#2}}{\@ACRlong{#1}{#2} []}%
2816 }
```

Read in the final optional argument:

```
2817 \def\@ACRlong#1#2[#3] {%
2818   \glsdoifexists{#2}%
2819   {%
2820     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2821   \protected@edef\@glo@text{\glsentrylong{#2}}%
```

Call \@gls@link

```
2822   \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2823   }%
2824 }
```

Short plural:

\acrlongpl

```
2825 \newrobustcmd*{\acrlongpl}{\@ifstar\s@acrlongpl\ns@acrlongpl}
```

Define the starred form:

```
2826 \newcommand*{\s@acrlongpl}[2] [] {%
2827   \new@ifnextchar[{\@acrlongpl{hyper=false,#1}{#2}}%
2828     {\@acrlongpl{hyper=false,#1}{#2} []}%
2829 }
```

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

```
2830 \newcommand*{\ns@acrlongpl}[2] [] {%
2831   \new@ifnextchar[{\@acrlongpl{#1}{#2}}{\@acrlongpl{#1}{#2} []}%
2832 }
```

Read in the final optional argument:

```
2833 \def\@acrlongpl#1#2[#3] {%
2834   \glsdoifexists{#2}%
2835   {%
2836     \edef\@glo@type{\glsentrytype{#2}}%
```

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

```
2837   \protected@edef\@glo@text{\glsentrylongpl{#2}}%
```

```

Call \@gls@link
2838   \@gls@link[#1]{#2}{\@glo@text#3}%
2839   }%
2840 }

```

\Acrlongpl

```

2841 \newrobustcmd*{\Acrlongpl}{\@ifstar\s@Acrlongpl\ns@Acrlongpl}

```

Define the starred form:

```

2842 \newcommand*{\s@Acrlongpl}[2] [] {%
2843   \new@ifnextchar[{\@Acrlongpl{hyper=false,#1}{#2}}%
2844   {\@Acrlongpl{hyper=false,#1}{#2} []}%
2845 }

```

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

```

2846 \newcommand*{\ns@Acrlongpl}[2] [] {%
2847   \new@ifnextchar[{\@Acrlongpl{#1}{#2}}{\@Acrlongpl{#1}{#2} []}%
2848 }

```

Read in the final optional argument:

```

2849 \def\@Acrlongpl#1#2[#3] {%
2850   \glsdoifexists{#2}%
2851   {%
2852     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

2853   \protected@edef\@glo@text{\glsentrylongpl{#2}}%

```

Call \@gls@link

```

2854   \@gls@link[#1]{#2}%
2855   {%
2856     \expandafter\makefirstuc\expandafter{\@glo@text}#3%
2857   }%
2858   }%
2859 }

```

\ACRlongpl

```

2860 \newrobustcmd*{\ACRlongpl}{\@ifstar\s@ACRlongpl\ns@ACRlongpl}

```

Define the starred form:

```

2861 \newcommand*{\s@ACRlongpl}[2] [] {%
2862   \new@ifnextchar[{\@ACRlongpl{hyper=false,#1}{#2}}%
2863   {\@ACRlongpl{hyper=false,#1}{#2} []}%
2864 }

```

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

```

2865 \newcommand*{\ns@ACRlongpl}[2] [] {%
2866   \new@ifnextchar[{\@ACRlongpl{#1}{#2}}{\@ACRlongpl{#1}{#2} []}%
2867 }

```

Read in the final optional argument:

```

2868 \def\@ACRlongpl#1#2[#3]{%
2869   \glsdoifexists{#2}%
2870   {%
2871     \edef\@glo@type{\glsentrytype{#2}}%
2872     \protected@edef\@glo@text{\glsentrylongpl{#2}}%
2873     \Call\@gls@link
2874     \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text#3}}%
2875   }%
2876 }

```

### 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
2876 \newcommand*{\glsentryname}[1]{\csname glo@#1@name\endcsname}

```

```

\Glsentryname
2877 \newcommand*{\Glsentryname}[1]{%
2878 \protected@edef\@glo@text{\csname glo@#1@name\endcsname}%
2879 \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
2880 \newcommand*{\glsentrydesc}[1]{\csname glo@#1@desc\endcsname}

```

```

\Glsentrydesc
2881 \newcommand*{\Glsentrydesc}[1]{%
2882 \protected@edef\@glo@text{\csname glo@#1@desc\endcsname}%
2883 \expandafter\makefirstuc\expandafter{\@glo@text}}

```

Plural form:

`\glentrydescplural`

```
2884 \newcommand*{\glentrydescplural}[1]{%
2885 \csname glo@#1@descplural\endcsname}
```

`\Glsentrydescplural`

```
2886 \newcommand*{\Glsentrydescplural}[1]{%
2887 \protected@edef\@glo@text{\csname glo@#1@descplural\endcsname}%
2888 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Get the entry text, as specified by the text key when the entry was defined.  
The argument is the label associated with the entry:

`\glentrytext`

```
2889 \newcommand*{\glentrytext}[1]{\csname glo@#1@text\endcsname}
```

`\Glsentrytext`

```
2890 \newcommand*{\Glsentrytext}[1]{%
2891 \protected@edef\@glo@text{\csname glo@#1@text\endcsname}%
2892 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Get the plural form:

`\glentryplural`

```
2893 \newcommand*{\glentryplural}[1]{\csname glo@#1@plural\endcsname}
```

`\Glsentryplural`

```
2894 \newcommand*{\Glsentryplural}[1]{%
2895 \protected@edef\@glo@text{\csname glo@#1@plural\endcsname}%
2896 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Get the symbol associated with this entry. The argument is the label associated with the entry. Note that unless you used `symbol=false` in the `sanitize` package option you may get unexpected results if the symbol key contained any commands.

`\glentrysymbol`

```
2897 \newcommand*{\glentrysymbol}[1]{\csname glo@#1@symbol\endcsname}
```

`\Glsentrysymbol`

```
2898 \newcommand*{\Glsentrysymbol}[1]{%
2899 \protected@edef\@glo@text{\csname glo@#1@symbol\endcsname}%
2900 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Plural form:

`\glentrysymbolplural`

```
2901 \newcommand*{\glentrysymbolplural}[1]{%
2902 \csname glo@#1@symbolplural\endcsname}
```

lentrysymbolplural

```
2903 \newcommand*{\Glsentrysymbolplural}[1]{%
2904 \protected@edef\@glo@text{\csname glo@#1@symbolplural\endcsname}%
2905 \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

```
2906 \newcommand*{\glsentryfirst}[1]{\csname glo@#1@first\endcsname}
```

\Glsentryfirst

```
2907 \newcommand*{\Glsentryfirst}[1]{%
2908 \protected@edef\@glo@text{\csname glo@#1@first\endcsname}%
2909 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

Get the plural form (as specified by the firstplural key when the entry was defined).

glsentryfirstplural

```
2910 \newcommand*{\glsentryfirstplural}[1]{%
2911 \csname glo@#1@firstpl\endcsname}
```

Glsentryfirstplural

```
2912 \newcommand*{\Glsentryfirstplural}[1]{%
2913 \protected@edef\@glo@text{\csname glo@#1@firstpl\endcsname}%
2914 \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

```
2915 \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

```
2916 \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.

```
2917 \newcommand*{\glsentryuseri}[1]{\csname glo@#1@useri\endcsname}
```

\Glsentryuseri

```
2918 \newcommand*{\Glsentryuseri}[1]{%
2919 \protected@edef\@glo@text{\csname glo@#1@useri\endcsname}%
2920 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

`\glentryuserii` Get the second user key (as specified by the user2 when the entry was defined).  
The argument is the label associated with the entry.

```
2921 \newcommand*{\glentryuserii}[1]{\csname glo@#1@userii\endcsname}
```

`\Glsentryuserii`

```
2922 \newcommand*{\Glsentryuserii}[1]{%
2923 \protected@edef\@glo@text{\csname glo@#1@userii\endcsname}%
2924 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

`\glentryuseriii` Get the third user key (as specified by the user3 when the entry was defined).  
The argument is the label associated with the entry.

```
2925 \newcommand*{\glentryuseriii}[1]{\csname glo@#1@useriii\endcsname}
```

`\Glsentryuseriii`

```
2926 \newcommand*{\Glsentryuseriii}[1]{%
2927 \protected@edef\@glo@text{\csname glo@#1@useriii\endcsname}%
2928 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

`\glentryuseriv` Get the fourth user key (as specified by the user4 when the entry was defined).  
The argument is the label associated with the entry.

```
2929 \newcommand*{\glentryuseriv}[1]{\csname glo@#1@useriv\endcsname}
```

`\Glsentryuseriv`

```
2930 \newcommand*{\Glsentryuseriv}[1]{%
2931 \protected@edef\@glo@text{\csname glo@#1@useriv\endcsname}%
2932 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

`\glentryuserv` Get the fifth user key (as specified by the user5 when the entry was defined).  
The argument is the label associated with the entry.

```
2933 \newcommand*{\glentryuserv}[1]{\csname glo@#1@userv\endcsname}
```

`\Glsentryuserv`

```
2934 \newcommand*{\Glsentryuserv}[1]{%
2935 \protected@edef\@glo@text{\csname glo@#1@userv\endcsname}%
2936 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

`\glentryuservi` Get the sixth user key (as specified by the user6 when the entry was defined).  
The argument is the label associated with the entry.

```
2937 \newcommand*{\glentryuservi}[1]{\csname glo@#1@uservi\endcsname}
```

`\Glsentryuservi`

```
2938 \newcommand*{\Glsentryuservi}[1]{%
2939 \protected@edef\@glo@text{\csname glo@#1@uservi\endcsname}%
2940 \expandafter\makefirstuc\expandafter{\@glo@text}}
```

`\glentryshort` Get the short key (as specified by the short the entry was defined). The argument is the label associated with the entry.

```
2941 \newcommand*{\glentryshort}[1]{\csname glo@#1@short\endcsname}
```

```

\Glsentryshort
2942 \newcommand*{\Glsentryshort}[1]{%
2943 \protected@edef\@glo@text{\csname glo@#1@short\endcsname}%
2944 \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.
2945 \newcommand*{\glsentryshortpl}[1]{\csname glo@#1@shortpl\endcsname}

\Glsentryshortpl
2946 \newcommand*{\Glsentryshortpl}[1]{%
2947 \protected@edef\@glo@text{\csname glo@#1@shortpl\endcsname}%
2948 \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.
2949 \newcommand*{\glsentrylong}[1]{\csname glo@#1@long\endcsname}

\Glsentrylong
2950 \newcommand*{\Glsentrylong}[1]{%
2951 \protected@edef\@glo@text{\csname glo@#1@long\endcsname}%
2952 \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.
2953 \newcommand*{\glsentrylongpl}[1]{\csname glo@#1@longpl\endcsname}

\Glsentrylongpl
2954 \newcommand*{\Glsentrylongpl}[1]{%
2955 \protected@edef\@glo@text{\csname glo@#1@longpl\endcsname}%
2956 \expandafter\makefirstuc\expandafter{\@glo@text}}

                Short cut macros to access full form:

\glsentryfull
2957 \newcommand*{\glsentryfull}[1]{%
2958   \glsentrylong{#1}\space(\glsentryshort{#1})%
2959 }

\Glsentryfull
2960 \newcommand*{\Glsentryfull}[1]{%
2961   \Glsentrylong{#1}\space(\glsentryshort{#1})%
2962 }

\glsentryfullpl
2963 \newcommand*{\glsentryfullpl}[1]{%
2964   \glsentrylongpl{#1}\space(\glsentryshortpl{#1})%
2965 }

```

```

\Glsentryfullpl
2966 \newcommand*{\Glsentryfullpl}[1]{%
2967   \Glsentrylongpl{#1}\space(\Glsentryshortpl{#1})%
2968 }

\glsentrynumberlist  Displays the number list as is.
2969 \newcommand*{\glsentrynumberlist}[1]{%
2970   \glsdoifexists{#1}%
2971   {%
2972     \csname glo@#1@numberlist\endcsname
2973   }%
2974 }

\glsdisplaynumberlist  Formats the number list for the given entry label. Doesn't work with hyperref.
2975 \@ifpackageloaded{hyperref}
2976 {%
2977   \newcommand*{\glsdisplaynumberlist}[1]{%
2978     \GlossariesWarning
2979     {%
2980       \string\glsdisplaynumberlist\space
2981       doesn't work with hyperref.^^JUsing
2982       \string\glsentrynumberlist\space instead%
2983     }%
2984     \glsentrynumberlist{#1}%
2985   }%
2986 }%
2987 {%
2988   \newcommand*{\glsdisplaynumberlist}[1]{%
2989     \glsdoifexists{#1}%
2990     {%
2991       \bgroup
2992       \def\@glo@label{#1}%
2993       \let\@org@glnumberformat\glnumberformat
2994       \def\glnumberformat##1{##1}%
2995       \protected@edef\the@numberlist{\csname glo@\@glo@label @numberlist\endcsname}%
2996       \def\@gls@numlist@sep{}%
2997       \def\@gls@numlist@nextsep{}%
2998       \def\@gls@numlist@lastsep{}%
2999       \def\@gls@thislist{}%
3000       \def\@gls@donext@def{}%
3001       \renewcommand\do[1]{%
3002         \protected@edef\@gls@thislist{%
3003           \@gls@thislist
3004           \noexpand\@gls@numlist@sep
3005           ##1%
3006         }%
3007         \let\@gls@numlist@sep\@gls@numlist@nextsep
3008         \def\@gls@numlist@nextsep{\glnumlistsep}%
3009         \@gls@donext@def

```

```

3010      \def\@gls@donext@def{%
3011      \def\@gls@numlist@lastsep{\glsnumlistlastsep}%
3012      }%
3013      }%
3014      \expandafter \glsnumlistparser \expandafter{\the@numberlist}%
3015      \let\@gls@numlist@sep\@gls@numlist@lastsep
3016      \@gls@thislist
3017      \egroup
3018      }%
3019      }
3020      }

```

`\glsnumlistsep`

```

3021 \newcommand*{\glsnumlistsep}{, }

```

`\glsnumlistlastsep`

```

3022 \newcommand*{\glsnumlistlastsep}{ \& }

```

`\glshyperlink` Provide a hyperlink to a glossary entry without adding information to the glossary file. The entry needs to be added using a command like `\glslink` or `\glsadd` to ensure that the target is defined. The first (optional) argument specifies the link text. The entry name is used by default. The second argument is the entry label.

```

3023 \newcommand*{\glshyperlink}[2][\glsentrytext{\@glo@label}]{%
3024 \def\@glo@label{#2}%
3025 \@glslink{\glo@linkprefix#2}{#1}}

```

## 1.11 Adding an entry to the glossary without generating text

The following keys are provided for `\glsadd` and `\glsaddall`:

```

3026 \define@key{glossadd}{counter}{\def\@gls@counter{#1}}
3027 \define@key{glossadd}{format}{\def\@glsnumberformat{#1}}

```

This key is only used by `\glsaddall`:

```

3028 \define@key{glossadd}{types}{\def\@glo@type{#1}}

```

`\glsadd[<options>]{<label>}`

Add a term to the glossary without generating any link text. The optional argument indicates which counter to use, and how to format it (using a key-value list) the second argument is the entry label. Note that *<options>* only has two keys: counter and format (the types key will be ignored).

`\glsadd`

```

3029 \newrobustcmd*{\glsadd}[2][]{%
3030   \glsdoifexists{#2}%
3031   {%

```

```

3032 \def\@glsnumberformat{glsnumberformat}%
3033 \edef\@gls@counter{\csname glo@#2@counter\endcsname}%
3034 \setkeys{glossadd}{#1}%

Store the entry's counter in \theglsentrycounter
3035 \@gls@saveentrycounter
3036 \@do@wrglossary{#2}%
3037 }%
3038 }

```

`\glsaddall` [*<option list>*]

Add all terms defined for the listed glossaries (without displaying any text). If types key is omitted, apply to all glossary types.

```

\glsaddall
3039 \newrobustcmd*{\glsaddall}[1] [] {%
3040 \edef\@glo@type{\@glo@types}%
3041 \setkeys{glossadd}{#1}%
3042 \forallglsentries[\@glo@type]{\@glo@entry}{%
3043 \glsadd[#1]{\@glo@entry}}%
3044 }

```

## 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.

```

3045 \edef\glsopenbrace{\expandafter\@gobble\string\{ }

```

`\glsclosebrace` Define `\glsclosebrace` to make it easier to write an opening brace to a file.

```

3046 \edef\glsclosebrace{\expandafter\@gobble\string\} }

```

`\glsquote` Define command that makes it easier to write quote marks to a file in the event that the double quote character has been made active.

```

3047 \edef\glsquote#1{\string"#1\string"}

```

`\@glsfirstletter` Define the first letter to come after the digits 0,...,9. Only required for xindy.

```

3048 \ifglsxindy
3049   \newcommand*{\@glsfirstletter}{A}
3050 \fi

```

`\stLetterAfterDigits` Sets the first letter to come after the digits 0,...,9.

```

3051 \ifglsxindy
3052   \newcommand*{\GlsSetXdyFirstLetterAfterDigits}[1]{%
3053     \renewcommand*{\@glsfirstletter}{#1}}
3054 \else
3055   \newcommand*{\GlsSetXdyFirstLetterAfterDigits}[1]{%
3056     \glsnoxywarning\GlsSetXdyFirstLetterAfterDigits}
3057 \fi

```

`\@glsminrange` Define the minimum number of successive location references to merge into a range.

```

3058 \newcommand*{\@glsminrange}{2}

```

`\setXdyMinRangeLength` Set the minimum range length. The value must either be none or a positive integer. The glossaries package doesn't check if the argument is valid, that is left to xindy.

```

3059 \ifglsxindy
3060   \newcommand*{\GlsSetXdyMinRangeLength}[1]{%
3061     \renewcommand*{\@glsminrange}{#1}}
3062 \else
3063   \newcommand*{\GlsSetXdyMinRangeLength}[1]{%
3064     \glsnoxywarning\GlsSetXdyMinRangeLength}
3065 \fi

```

`\writeist`

```

3066 \ifglsxindy
    Code to use if xindy is required.
3067   \def\writeist{%
    Update attributes list
3068     \@gls@addpredefinedattributes
    Open the file.
3069     \openout\glswrite=\istfilename
    Write header comment at the start of the file
3070     \write\glswrite{;; xindy style file created by the glossaries
3071       package}%
3072     \write\glswrite{;; for document '\jobname' on
3073       \the\year-\the\month-\the\day}%

```

Specify the required styles

```

3074 \write\glswrite{^^J; required styles^^J}
3075 \@for\@xdystyle:=\@xdyrequiredstyles\do{%
3076     \ifx\@xdystyle\@empty
3077     \else
3078     \protected@write\glswrite{}\{(require
3079     \string"\@xdystyle.xdy\string")}%
3080     \fi
3081 }%

```

List the allowed attributes (possible values used by the format key)

```

3082 \write\glswrite{^^J%
3083     ; list of allowed attributes (number formats)^^J}%
3084 \write\glswrite{(define-attributes ((\@xdyattributes)))}%

```

Define any additional alphabets

```

3085 \write\glswrite{^^J; user defined alphabets^^J}%
3086 \write\glswrite{\@xdyuseralphabets}%

```

Define location classes.

```

3087 \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.

```

3088 \@for\@gls@classI:=\@gls@xdy@locationlist\do{%

```

Case were  $\langle Hprefix \rangle$  is empty:

```

3089 \protected@write\glswrite{}\{(define-location-class
3090 \string"\@gls@classI\string"^^J\space\space\space
3091 (
3092     :sep "{}{"
3093     \csname @gls@xdy@Lclass@\@gls@classI\endcsname\space
3094     :sep "}"
3095 )
3096 ^^J\space\space\space
3097 :min-range-length \@glsminrange^^J%
3098 )
3099 }%

```

Nested iteration over all classes:

```

3100 {%
3101     \@for\@gls@classII:=\@gls@xdy@locationlist\do{%
3102         \protected@write\glswrite{}\{(define-location-class
3103         \string"\@gls@classII-\@gls@classI\string"
3104         ^^J\space\space\space
3105         (
3106             :sep "{"
3107             \csname @gls@xdy@Lclass@\@gls@classII\endcsname\space
3108             :sep "{}{"
3109             \csname @gls@xdy@Lclass@\@gls@classI\endcsname\space
3110             :sep "}"

```

```

3111         )
3112         ^^J\space\space\space
3113         :min-range-length \@glsminrange^^J%
3114     )
3115 }%
3116 }%
3117 }%
3118 }%

```

User defined location classes (needs checking for new location format).

```

3119 \write\glswrite{^^J; user defined location classes}%
3120 \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.)

```

3121 \write\glswrite{^^J; define cross-reference class^^J}%
3122 \write\glswrite{(define-crossref-class \string"see\string"
3123 :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.)

```

3124 \write\glswrite{(markup-crossref-list
3125 :class \string"see\string"^^J\space\space\space
3126 :open \string"\string\glsseeformat\string"
3127 :close \string"{}\string")}%

```

List the order to sort the classes.

```

3128 \write\glswrite{^^J; define the order of the location classes}%
3129 \write\glswrite{(define-location-class-order
3130 (\@xdylocationclassorder))}%

```

Specify what to write to the start and end of the glossary file.

```

3131 \write\glswrite{^^J; define the glossary markup^^J}%

3132 \write\glswrite{(markup-index^^J\space\space\space
3133 :open \string"\string
3134 \glossarysection[\string\glossarytoctitle]{\string
3135 \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)

```

3136 \@for\@this@ctr:=\@xdycounters\do{%
3137     {%
3138         \@for\@this@attr:=\@xdyattributelist\do{%
3139             \protected@write\glswrite{}{\string\providecommand*%
3140                 \expandafter\string
3141                 \csname glsX\@this@ctr X\@this@attr\endcsname[2]%
3142                 {%

```

```

3143         \string\setentrycounter
3144         [\expandafter\@gobble\string\#1]{\@this@ctr}%
3145         \expandafter\string
3146         \csname\@this@attr\endcsname
3147         {\expandafter\@gobble\string\#2}%
3148     }%
3149 }%
3150 }%
3151 }%
3152 }%

```

Add the end part of the open tag and the rest of the markup-index information:

```

3153 \write\glswrite{%
3154     \string\begin
3155     {theglossary}\string\glossaryheader\string~n\string" ^^J\space
3156     \space\space:close \string"\expandafter\@gobble
3157     \string%\string~n\string
3158     \end{theglossary}\string\glossarypostamble
3159     \string~n\string" ^^J\space\space\space
3160     :tree)}}%

```

Specify what to put between letter groups

```

3161 \write\glswrite{(markup-letter-group-list
3162     :sep \string"\string\glsgroupskip\string~n\string"))}%

```

Specify what to put between entries

```

3163 \write\glswrite{(markup-indexentry
3164     :open \string"\string\relax \string\glsresetentrylist
3165     \string~n\string"))}%

```

Specify how to format entries

```

3166 \write\glswrite{(markup-locclass-list :open
3167     \string"\glsopenbrace\string\glossaryentrynumbers
3168     \glsopenbrace\string\relax\space \string"^^J\space\space\space
3169     :sep \string", \string"
3170     :close \string"\glsclosebrace\glsclosebrace\string"))}%

```

Specify how to separate location numbers

```

3171 \write\glswrite{(markup-locref-list
3172     :sep \string"\string\delimN\space\string"))}%

```

Specify how to indicate location ranges

```

3173 \write\glswrite{(markup-range
3174     :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.

```

3175 \@onelevel@sanitize\gls@suffixF
3176 \@onelevel@sanitize\gls@suffixFF
3177 \ifx\gls@suffixF\@empty
3178 \else

```

```

3179     \write\glswrite{(markup-range
3180       :close "\gls@suffixF" :length 1 :ignore-end)}}%
3181   \fi
3182   \ifx\gls@suffixFF\@empty
3183   \else
3184     \write\glswrite{(markup-range
3185       :close "\gls@suffixFF" :length 2 :ignore-end)}}%
3186   \fi

```

Specify how to format locations.

```

3187   \write\glswrite{^^J; define format to use for locations^^J}%
3188   \write\glswrite{\@xdylocref}%

```

Specify how to separate letter groups.

```

3189   \write\glswrite{^^J; define letter group list format^^J}%
3190   \write\glswrite{(markup-letter-group-list
3191     :sep \string"\string\glsgroupskip\string~n\string")}%

```

Define letter group headings.

```

3192   \write\glswrite{^^J; letter group headings^^J}%
3193   \write\glswrite{(markup-letter-group
3194     :open-head \string"\string\glsgroupheading
3195     \glsopenbrace\string"^^J\space\space\space
3196     :close-head \string"\glsclosebrace\string")}%

```

Define additional letter groups.

```

3197   \write\glswrite{^^J; additional letter groups^^J}%
3198   \write\glswrite{\@xdylettergroups}%

```

Define additional sort rules

```

3199   \write\glswrite{^^J; additional sort rules^^J}
3200   \write\glswrite{\@xdysortrules}%

```

Close the style file

```

3201   \closeout\glswrite

```

Suppress any further calls.

```

3202   \let\writeist\relax
3203 }
3204 \else

```

Code to use if makeindex is required.

```

3205 \edef\@gls@actualchar{\string?}
3206 \edef\@gls@encapchar{\string|}
3207 \edef\@gls@levelchar{\string!}
3208 \edef\@gls@quotechar{\string"}
3209 \def\writeist{\relax
3210   \openout\glswrite=\istfilename
3211   \write\glswrite{\expandafter\@gobble\string\% makeindex style file
3212     created by the glossaries package}
3213   \write\glswrite{\expandafter\@gobble\string\% for document
3214     '\jobname' on \the\year-\the\month-\the\day}

```

```

3215 \write\glswrite{actual '\@gls@actualchar'}
3216 \write\glswrite{encap '\@gls@encapchar'}
3217 \write\glswrite{level '\@gls@levelchar'}
3218 \write\glswrite{quote '\@gls@quotechar'}
3219 \write\glswrite{keyword \string"\string\glossaryentry\string"}
3220 \write\glswrite{preamble \string"\string\glossarysection[\string
3221 \glossarytoctitle]{\string\glossarytitle}\string
3222 \glossarypreamble\string\n\string\begin{theglossary}\string
3223 \glossaryheader\string\n\string"}
3224 \write\glswrite{postamble \string"\string%\string\n\string
3225 \end{theglossary}\string\glossarypostamble\string\n
3226 \string"}
3227 \write\glswrite{group_skip \string"\string\glsgroupskip\string\n
3228 \string"}
3229 \write\glswrite{item_0 \string"\string%\string\n\string"}
3230 \write\glswrite{item_1 \string"\string%\string\n\string"}
3231 \write\glswrite{item_2 \string"\string%\string\n\string"}
3232 \write\glswrite{item_01 \string"\string%\string\n\string"}
3233 \write\glswrite{item_x1
3234 \string"\string\relax \string\glsresetentrylist\string\n
3235 \string"}
3236 \write\glswrite{item_12 \string"\string%\string\n\string"}
3237 \write\glswrite{item_x2
3238 \string"\string\relax \string\glsresetentrylist\string\n
3239 \string"}

3240 \write\glswrite{delim_0 \string"\string\{\string
3241 \glossaryentrynumbers\string\{\string\relax \string"}
3242 \write\glswrite{delim_1 \string"\string\{\string
3243 \glossaryentrynumbers\string\{\string\relax \string"}
3244 \write\glswrite{delim_2 \string"\string\{\string
3245 \glossaryentrynumbers\string\{\string\relax \string"}
3246 \write\glswrite{delim_t \string"\string\}\string\}\string"}
3247 \write\glswrite{delim_n \string"\string\delimN \string"}
3248 \write\glswrite{delim_r \string"\string\delimR \string"}
3249 \write\glswrite{headings_flag 1}
3250 \write\glswrite{heading_prefix
3251 \string"\string\glsgroupheading\string\{\string"}
3252 \write\glswrite{heading_suffix
3253 \string"\string\}\string\relax
3254 \string\glsresetentrylist \string"}
3255 \write\glswrite{symhead_positive \string"glssymbols\string"}
3256 \write\glswrite{numhead_positive \string"glssymbols\string"}
3257 \write\glswrite{page_compositor \string"glscpositor\string"}
3258 \@gls@escbsdq\gls@suffixF
3259 \@gls@escbsdq\gls@suffixFF
3260 \ifx\gls@suffixF\empty
3261 \else
3262 \write\glswrite{suffix_2p \string"\gls@suffixF\string"}
3263 \fi

```

```

3264 \ifx\gls@suffixFF\@empty
3265 \else
3266 \write\glswrite{suffix_3p \string"\gls@suffixFF\string"}
3267 \fi
3268 \closeout\glswrite
3269 \let\writeist\relax
3270 }
3271 \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`

```

3272 \newcommand{\noist}{%
  Update attributes list
3273 \@gls@addpredefinedattributes
3274 \let\writeist\relax
3275 }

```

`\@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`

```

3276 \newcommand*{\@makeglossary}[1]{%
3277 \ifglossaryexists{#1}%
3278 {%

```

Only create a new write if `savewrites=false` otherwise create a token to collect the information.

```

3279 \ifglssavewrites
3280 \expandafter\newtoks\csname glo@#1@filetok\endcsname
3281 \else
3282 \expandafter\newwrite\csname glo@#1@file\endcsname
3283 \expandafter\@glsopenfile\csname glo@#1@file\endcsname{#1}%
3284 \fi
3285 \@gls@renewglossary
3286 \writeist
3287 }%
3288 {%
3289 \PackageError{glossaries}%

```

```

3290    {Glossary type ‘#1’ not defined}%
3291    {New glossaries must be defined before using \string\makeglossary}%
3292  }%
3293 }

```

`\@glsopenfile` Open write file associated with the given glossary.

```

3294 \newcommand*{\@glsopenfile}[2]{%
3295   \immediate\openout#1=\jobname.\csname @glotype@#2@out\endcsname
3296   \PackageInfo{glossaries}{Writing glossary file
3297     \jobname.\csname @glotype@#2@out\endcsname}%
3298 }

```

`\nomakeglossaries` Issue warning that `\makeglossaries` hasn't been used.

```

3299 \newcommand*{\warn@nomakeglossaries}{%
3300   \GlossariesWarningNoLine{\string\makeglossaries\space
3301     hasn't been used,^^Jthe glossaries will not be updated}%
3302 }

```

`\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`

```

3303 \newcommand*{\makeglossaries}{%
3304   Write the name of the style file to the aux file (needed by makeglossaries)
3305   \protected@write\@auxout{}{\string\@istfilename{\istfilename}}%
3306   \protected@write\@auxout{}{\string\@glsorder{\glsorder}}%
3307   Iterate through each glossary type and activate it.
3308   \@for\@glo@type:=\@glo@types\do{%
3309     \ifthenelse{\equal{\@glo@type}{}}{}{}%
3310     \@makeglossary{\@glo@type}%
3311   }%

```

New glossaries must be created before `\makeglossaries` so disable `\newglossary`.

```

3310 \renewcommand*\newglossary[4][]{%
3311   \PackageError{glossaries}{New glossaries
3312     must be created before \string\makeglossaries}{You need
3313     to move \string\makeglossaries\space after all your
3314     \string\newglossary\space commands}}%

```

Any subsequent instances of this command should have no effect

```

3315 \let\@makeglossary\relax
3316 \let\makeglossary\relax
3317 \let\makeglossaries\relax

```

Disable all commands that have no effect after `\makeglossaries`

```

3318 \@disable@onlypremakeg

```

Suppress warning about no \makeglossaries

```

3319 \let\warn@nomakeglossaries\relax

Declare list parser for \glsdisplaynumberlist
3320 \ifglssavenumberlist
3321   \edef\@gls@dodolistparser{\noexpand\DeclareListParser
3322     {\noexpand\glsnumlistparser}{\delimN}}}%
3323   \@gls@dodolistparser
3324 \fi
3325 }

```

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

```

3326 \let\makeglossary\makeglossaries

If \makeglossaries hasn't been used, issue a warning. Also issue a warning
if neither \printglossaries nor \printglossary have been used.
3327 \AtEndDocument{%
3328   \warn@nomakeglossaries
3329   \warn@noprintglossary
3330 }

```

### 1.13 Writing information to associated files

\glswrite The write used for style file also used for all other output files if savewrites=true.

```

3331 \newwrite\glswrite

```

\istfile Deprecated.

```

3332 \def\istfile{\glswrite}

```

At the end of the document, the files should be created if savewrites=true.

```

3333 \AtEndDocument{%
3334   \glswritefiles
3335 }

```

\glswritefiles Only write the files if savewrites=true

```

3336 \ifglssavewrites
3337   \newcommand*{\glswritefiles}{%

```

Iterate through all the glossaries

```

3338   \forallglossaries{\@glo@type}{%

```

Check for empty glossaries (patch provided by Patrick Häcker)

```

3339     \ifcsundef{glo@\@glo@type @filetok}%
3340     {%
3341       \def\gls@tmp{%

```

```

3342     }%
3343     {%
3344         \edef\gls@tmp{\expandafter\the
3345             \csname glo@\@glo@type @filetok\endcsname}%
3346     }%
3347     \ifx\gls@tmp\@empty
3348         \ifx\@glo@type\glsdefaulttype
3349             \GlossariesWarningNoLine{Glossary '@glo@type' has no
3350                 entries.^^JRemember to use package option 'nomain' if
3351 you
3352                 don't want to^^Juse the main glossary}%
3353         \else
3354             \GlossariesWarningNoLine{Glossary '@glo@type' has no
3355                 entries}%
3356         \fi
3357     \else
3358         \@glsopenfile{\glswrite}{\@glo@type}%
3359         \immediate\write\glswrite{%
3360             \expandafter\the
3361             \csname glo@\@glo@type @filetok\endcsname}%
3362         \immediate\closeout\glswrite
3363     \fi
3364 }%
3365 }
3366 \else
3367     \let\glswritefiles\relax
3368 \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`

```

3369 \renewcommand*{\glossary}[1][\glsdefaulttype]{%
3370     \@glossary[#1]%
3371 }

```

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`

```

3372 \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

```

3373 \newcommand{\@gls@renewglossary}{%
3374   \gdef\@glossary[##1]{\@bsphack\begingroup\@wrglossary{##1}}%
3375   \let\@gls@renewglossary\@empty
3376 }
```

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

```

3377 \renewcommand*{\@wrglossary}[2]{%
3378   \ifglssavewrites
3379     \protected@edef\@gls@tmp{\the\csname glo@#1@filetok\endcsname#2}%
3380     \expandafter\global\expandafter\csname glo@#1@filetok\endcsname
3381       \expandafter{\@gls@tmp^^J}%
3382   \else
3383     \ifcsdef{glo@#1@file}%
3384     {%
3385       \expandafter\protected@write\csname glo@#1@file\endcsname{%
3386         \gls@disablepagerefexpansion}{#2}%
3387     }%
3388     {%
3389       \GlossariesWarning{No file defined for glossary ‘#1’}%
3390     }%
3391   \fi
3392   \endgroup\@esphack
3393 }
```

\@do@wrglossary

```

3394 \newcommand*{\@do@wrglossary}[1]{%
3395   \ifglsindexonlyfirst
3396     \ifglsused{#1}{\@do@wrglossary{#1}}%
3397   \else
3398     \@do@wrglossary{#1}%
3399   \fi
3400 }
```

@protected@pagefmts List of page formats to be protected against expansion.

```

3401 \newcommand{\gls@protected@pagefmts}{%
3402   \gls@numberpage,\gls@alphpage,\gls@Alphpage,\gls@romanpage,\gls@Romanpage%
3403 }
```

blepagerefexpansion

```

3404 \newcommand*{\gls@disablepagerefexpansion}{%
3405   \@for\@gls@this:=\gls@protected@pagefmts\do
3406   {%
3407     \expandafter\let\@gls@this\relax
```

```

3408 }%
3409 }

```

`\gls@alphpage`

```

3410 \newcommand*{\gls@alphpage}{\@alph\c@page}

```

`\gls@Alphpage`

```

3411 \newcommand*{\gls@Alphpage}{\@Alph\c@page}

```

`\gls@numberpage`

```

3412 \newcommand*{\gls@numberpage}{\number\c@page}

```

`\gls@romanpage`

```

3413 \newcommand*{\gls@romanpage}{\romannumeral\c@page}

```

`\gls@Romanpage`

```

3414 \newcommand*{\gls@Romanpage}{\@Roman\c@page}

```

`\@do@wrglossary` Write the glossary entry in the appropriate format. (Need to set `\@glsnumberformat` and `\@gls@counter` prior to use.) The argument is the entry's label.

```

3415 \newcommand*{\@do@wrglossary}[1]{%
3416   \begingroup

```

First a bit of hackery to prevent premature expansion of `\c@page`. Store original definitions:

```

3417   \let\orgthe\the
3418   \let\orgnumber\number
3419   \let\orgromannumeral\romannumeral
3420   \let\orgalph\@alph
3421   \let\orgAlph\@Alph
3422   \let\orgRoman\@Roman

```

Redefine:

```

3423   \def\the##1{%
3424     \ifx##1\c@page \gls@numberpage\else\orgthe##1\fi}%
3425   \def\number##1{%
3426     \ifx##1\c@page \gls@numberpage\else\orgnumber##1\fi}%
3427   \def\romannumeral##1{%
3428     \ifx##1\c@page \gls@romanpage\else\orgromannumeral##1\fi}%
3429   \def\@Roman##1{%
3430     \ifx##1\c@page \gls@Romanpage\else\orgRoman##1\fi}%
3431   \def\@alph##1{%
3432     \ifx##1\c@page \gls@alphpage\else\orgalph##1\fi}%
3433   \def\@Alph##1{%
3434     \ifx##1\c@page \gls@Alphpage\else\orgAlph##1\fi}%

```

Prevent expansion:

```

3435   \gls@disablepagerefexpansion

```

Now store location in \@glslocref:

```
3436 \protected@xdef\@glslocref{\theHglentrycounter}%  
3437 \endgroup
```

Escape any special characters

```
3438 \@gls@checkmkidxchars\@glslocref
```

Check if the hyper-location is the same as the location and set the hyper prefix.

```
3439 \expandafter\ifx\theHglentrycounter\theHglentrycounter  
3440 \def\@glo@counterprefix{%  
3441 \else  
3442 \protected@edef\@glsHlocref{\theHglentrycounter}%  
3443 \@gls@checkmkidxchars\@glsHlocref  
3444 \edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix  
3445 {\@glslocref}{\@glsHlocref}}%  
3446 }%  
3447 \@do@gls@getcounterprefix  
3448 \fi
```

Determine whether to use xindy or makeindex syntax

```
3449 \ifglxindy
```

Need to determine if the formatting information starts with a ( or ) indicating a range.

```
3450 \expandafter\@glo@check@mkidxrangechar\@glsnumberformat\@nil  
3451 \def\@glo@range{%  
3452 \expandafter\if\@glo@prefix(\relax  
3453 \def\@glo@range{:open-range}%  
3454 \else  
3455 \expandafter\if\@glo@prefix)\relax  
3456 \def\@glo@range{:close-range}%  
3457 \fi  
3458 \fi
```

Write to the glossary file using xindy syntax.

```
3459 \glossary[\csname glo@#1@type\endcsname]{%  
3460 (indexentry :tkey (\csname glo@#1@index\endcsname)  
  
3461 :locref \string{\@glo@counterprefix}{\@glslocref}\string" %  
3462 :attr \string"\@gls@counter\@glo@suffix\string"  
3463 \@glo@range  
3464 )  
3465 }%  
3466 \else
```

Convert the format information into the format required for makeindex

```
3467 \@set@glo@numformat{\@glo@numfmt}{\@gls@counter}{\@glsnumberformat}%  
3468 {\@glo@counterprefix}%
```

Write to the glossary file using makeindex syntax.

```
3469 \glossary[\csname glo@#1@type\endcsname]{%
```

```

3470 \string\glossaryentry{\csname glo@#1@index\endcsname
3471 \@gls@encapchar\@glo@numfmt}\@gls@locref}}}%
3472 \fi
3473 }

```

`\ls@getcounterprefix` Get the prefix that needs to be prepended to counter in order to get the hyper counter. (For example, with the standard article class and hyperref, `\theequation` needs to be prefixed with `\section num` to get the equivalent `\theHequation`.) NB this assumes that the prefix ends with a dot, which is the standard. (Otherwise it makes the xindy location classes more complicated.)

```

3474 \newcommand*\@gls@getcounterprefix[2]{%
3475 \edef\@gls@thisloc{#1}\edef\@gls@thisHloc{#2}%
3476 \ifx\@gls@thisloc\@gls@thisHloc
3477 \def\@glo@counterprefix{}%
3478 \else
3479 \def\@gls@get@counterprefix##1.#1##2\end@getprefix{%
3480 \def\@glo@tmp{##2}%
3481 \ifx\@glo@tmp\@empty
3482 \def\@glo@counterprefix{}%
3483 \else
3484 \def\@glo@counterprefix{##1}%
3485 \fi
3486 }%
3487 \@gls@get@counterprefix#2.#1\end@getprefix
3488 \fi
3489 }

```

## 1.14 Glossary Entry Cross-References

`\@do@seeglossary` Write the glossary entry with a cross reference. The first argument is the entry's label, the second must be in the form `[\<tag>]{\<list>}`, where `<tag>` is a tag such as “see” and `<list>` is a list of labels.

```

3490 \newcommand{\@do@seeglossary}[2]{%
3491 \def\@gls@xref{#2}%
3492 \@onelevel@sanitize\@gls@xref
3493 \@gls@checkmkidxchars\@gls@xref
3494 \ifglxsindy
3495 \glossary[\csname glo@#1@type\endcsname]{%
3496 (indexentry
3497 :tkey (\csname glo@#1@index\endcsname)
3498 :xref (\string"\@gls@xref\string")
3499 :attr \string"see\string"
3500 )
3501 }%
3502 \else
3503 \glossary[\csname glo@#1@type\endcsname]{%
3504 \string\glossaryentry{\csname glo@#1@index\endcsname
3505 \@gls@encapchar glsseeformat\@gls@xref}{Z}}}%

```

```

3506 \fi
3507 }

```

`\@gls@fixbraces` If no optional argument is specified, list needs to be enclosed in a set of braces.

```

3508 \def\@gls@fixbraces#1#2#3\@nil{%
3509   \ifx#2[\relax
3510     \def#1{#2#3}%
3511   \else
3512     \def#1{{#2#3}}%
3513   \fi
3514 }

```

`\glssee` `\glssee{<label>}{<cross-ref list>}`

```

3515 \newcommand*\glssee{3}[\seename]{%
3516   \@do@seeglossary{#2}{[#1]{#3}}
3517 \newcommand*\@glssee{3}[\seename]{%
3518   \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.

```

3519 \newcommand*\glsseeformat{3}[\seename]{\emph{#1} \glsseelist{#2}}

```

`\glsseelist` `\glsseelist{<list>}` formats list of entry labels.

```

3520 \newcommand*\glsseelist{1}{%

```

If there is only one item in the list, set the last separator to do nothing.

```

3521   \let\@gls@dolast\relax

```

Don’t display separator on the first iteration of the loop

```

3522   \let\@gls@donext\relax

```

Iterate through the labels

```

3523   \@for\@gls@thislabel:=#1\do{%

```

Check if on last iteration of loop

```

3524     \ifx\@xfor@nextelement\@nnil
3525       \@gls@dolast
3526     \else
3527       \@gls@donext
3528     \fi

```

display the entry for this label

```

3529     \glsseeitem{\@gls@thislabel}%

```

Update separators

```

3530     \let\@gls@dolast\glsseelastsep
3531     \let\@gls@donext\glsseesep
3532   }%
3533 }

```

`\glsseelastsep` Separator to use between penultimate and ultimate entries in a cross-referencing list.

```
3534 \newcommand*{\glsseelastsep}{\space\andname\space}
```

`\glsseesep` Separator to use between entires in a cross-referencing list.

```
3535 \newcommand*{\glsseesep}{, }
```

`\glsseeitem` `\glsseeitem{<label>}` formats individual entry in a cross-referencing list.

```
3536 \newcommand*{\glsseeitem}[1]{\glshyperlink[\glsseeitemformat{#1}]{#1}}
```

`\glsseeitemformat` As from v3.0, default is to use `\glsentrytext` instead of `\glsentryname`. (To avoid problems with the name key being sanitized.)

```
3537 \newcommand*{\glsseeitemformat}[1]{\glsentrytext{#1}}
```

## 1.15 Displaying the glossary

An individual glossary is displayed in the text using `\printglossary[<key-val list>]`. If the type key is omitted, the default glossary is displayed. The optional argument can be used to specify an alternative glossary, and can also be used to set the style, title and entry in the table of contents. Available keys are defined below.

`\gls@save@numberlist` Provide command to store number list.

```
3538 \newcommand*{\gls@save@numberlist}[1]{%
3539   \ifglssavenumberlist
3540     \toks@{#1}%
3541     \edef\@do@writeaux@info{%
3542       \noexpand\csgdef{glo@\glscurrententrylabel @numberlist}{\the\toks@}%
3543     }%
3544     \@onelevel@sanitize\@do@writeaux@info
3545     \protected@write\@auxout{}\@do@writeaux@info}%
3546   \fi
3547 }
```

`\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.)

```
3548 \def\warn@noprintglossary{%
3549   \GlossariesWarningNoLine{No \string\printglossary\space
3550     or \string\printglossaries\space
3551     found.^^JThis document will not have a glossary}%
3552 }
```

`\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.

```
3553 \ifcsundef{printglossary}{}%
3554 {%
```

If `\printglossary` is already defined, issue a warning and undefine it.

```

3555 \GlossariesWarning{Overriding \string\printglossary}%
3556 \undef\printglossary
3557 }

```

`\printglossary` has an optional argument. The default value is to set the glossary type to the main glossary.

```

3558 \newcommand*{\printglossary}[1][type=\glsdefaulttype]{%

```

Set up defaults.

```

3559 \def\@glo@type{\glsdefaulttype}%
3560 \def\glossarytitle{\csname @glo@type\@glo@type @title\endcsname}%

```

```

3561 \def\glossarytoctitle{\glossarytitle}%
3562 \let\org@glossarytitle\glossarytitle
3563 \def\@glossarystyle{}%
3564 \def\gls@dotoc@title{\glssettoctitle{\@glo@type}}%

```

Store current value of `\glossaryentrynumbers`. (This may be changed via the optional argument)

```

3565 \let\@org@glossaryentrynumbers\glossaryentrynumbers

```

Localise the effects of the optional argument

```

3566 \bgroup

```

Determine settings specified in the optional argument.

```

3567 \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)

```

3568 \ifx\glossarytitle\org@glossarytitle
3569 \else
3570 \expandafter\let\csname @glo@type\@glo@type @title\endcsname
3571 \glossarytitle
3572 \fi

```

Allow a high-level user command to indicate the current glossary

```

3573 \let\currentglossary\@glo@type

```

Enable individual number lists to be suppressed.

```

3574 \let\org@glossaryentrynumbers\glossaryentrynumbers
3575 \let\glsnonextpages\@glsnonextpages

```

Enable individual number list to be activated:

```

3576 \let\glsnextpages\@glsnextpages

```

Enable suppression of description terminators.

```

3577 \let\nopostdesc\@nopostdesc

```

Set up the entry for the TOC

```

3578 \gls@dotoc@title

```

Set the glossary style

```

3579 \@glossarystyle

```

added a way to fetch the current entry label:

```

3580 \let\gls@org@glossaryentryfield\glossaryentryfield
3581 \let\gls@org@glossarysubentryfield\glossarysubentryfield
3582 \renewcommand{\glossaryentryfield}[1]{%
3583   \gdef\glscurrententrylabel{##1}%
3584   \gls@org@glossaryentryfield{##1}%
3585 }%
3586 \renewcommand{\glossarysubentryfield}[2]{%
3587   \gdef\glscurrententrylabel{##2}%
3588   \gls@org@glossarysubentryfield{##1}{##2}%
3589 }%
```

Some macros may end up being expanded into internals in the glossary, so need to make @ a letter.

```

3590 \makeatletter
```

Input the glossary file, if it exists.

```

3591 \@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.

```

3592 \IfFileExists{\jobname.\csname @glo@type@\glo@type @in\endcsname}%
3593 {}%
3594 {\null}%
```

If xindy is being used, need to write the language dependent information to the .aux file for makeglossaries.

```

3595 \ifglxindy
3596   \ifcsundef{@xdy@\glo@type @language}%
3597   {%
3598     \edef\@do@auxoutstuff{%
3599       \noexpand\AtEndDocument{%
3600         \noexpand\immediate\noexpand\write\@auxout{%
3601           \string\@xdylanguage{\glo@type}{\@xdy@main@language}}%
3602       }%
3603     }%
3604   }%
3605   {%
3606     \edef\@do@auxoutstuff{%
3607       \noexpand\AtEndDocument{%
3608         \noexpand\immediate\noexpand\write\@auxout{%
3609           \string\@xdylanguage{\glo@type}{\csname @xdy@\glo@type
3610             @language\endcsname}}%
3611       }%
3612     }%
3613   }%
3614   \@do@auxoutstuff
3615   \edef\@do@auxoutstuff{%
3616     \noexpand\AtEndDocument{%
```

```

3617         \noexpand\immediate\noexpand\write\@auxout{%
3618         \string\@gls@codepage{\@glo@type}{\gls@codepage}}%
3619     }%
3620 }%
3621 \do@auxoutstuff
3622 \fi
3623 \egroup

Reset \glossaryentrynumbers
3624 \global\let\glossaryentrynumbers\@org@glossaryentrynumbers

Suppress warning about no \printglossary
3625 \global\let\warn@noprintglossary\relax
3626 }

```

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`

```

3627 \newcommand*{\printglossaries}{%
3628   \forallglossaries{\@glo@type}{\printglossary[type=\@glo@type]}%
3629 }

```

The keys that can be used in the optional argument to `\printglossary` are as follows: The `type` key sets the glossary type.

```

3630 \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`.

```

3631 \define@key{printgloss}{title}{%
3632   \def\glossarytitle{#1}%
3633   \let\gls@dotoc@title\relax
3634 }

```

The `toctitle` sets the text used for the relevant entry in the table of contents.

```

3635 \define@key{printgloss}{toctitle}{%
3636   \def\glossarytoctitle{#1}%
3637   \let\gls@dotoc@title\relax
3638 }

```

The `style` key sets the glossary style (but only for the given glossary).

```

3639 \define@key{printgloss}{style}{%
3640   \ifcsundef{\glsstyle@#1}%
3641   {%

```

```

3642 \PackageError{glossaries}%
3643 {Glossary style ‘#1’ undefined}{}%
3644 }%
3645 {%
3646 \def\@glossarystyle{\csname @glsstyle@#1\endcsname}%
3647 }%
3648 }

```

The `numberedsection` key determines if this glossary should be in a numbered section.

```

3649 \define@choicekey{printgloss}{numberedsection}[\val\nr]{%
3650 false,nolabel,autolabel}[nolabel]{%
3651 \ifcase\nr\relax
3652 \renewcommand*\@glossarysecstar{*}%
3653 \renewcommand*\@glossaryseclabel{}%
3654 \or
3655 \renewcommand*\@glossarysecstar{}%
3656 \renewcommand*\@glossaryseclabel{}%
3657 \or
3658 \renewcommand*\@glossarysecstar{}%
3659 \renewcommand*\@glossaryseclabel{\label{\glsautoprefix\@glo@type}}%
3660 \fi}

```

The `nonumberlist` key determines if this glossary should have a number list.

```

3661 \define@boolkey{printgloss}[gls]{nonumberlist}[true]{%
3662 \ifglsnonumberlist
3663 \def\glossaryentrynumbers##1{}%
3664 \else
3665 \def\glossaryentrynumbers##1{##1}%
3666 \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 placed in the entry’s description and 3 column tabular style glossary is used.) `\org@glossaryentrynumbers` needs to be set at the start of each glossary, in the event that `\glossaryentrynumber` is redefined.

```

3667 \newcommand*\@glsnonextpages{%
3668 \gdef\glossaryentrynumbers##1{%
3669 \glsresetentrylist
3670 }%
3671 }

```

`\@glsnextpages` Activate the next number list only. Global assignments required as it may not occur in the same level of grouping as the next numberlist. (For example, if `\glsnextpages` is placed in the entry’s description and 3 column tabular style glossary is used.) `\org@glossaryentrynumbers` needs to be set at the start of each glossary, in the event that `\glossaryentrynumber` is redefined.

```

3672 \newcommand*\@glsnextpages{%
3673 \gdef\glossaryentrynumbers##1{%

```

```

3674      ##1\glsresetentrylist}}

\glsresetentrylist  Resets \glossaryentrynumbers
3675 \newcommand*{\glsresetentrylist}{%
3676   \global\let\glossaryentrynumbers\org@glossaryentrynumbers}

\glsnonextpages  Outside of \printglossary this does nothing.
3677 \newcommand*{\glsnonextpages}{}

\glsnextpages  Outside of \printglossary this does nothing.
3678 \newcommand*{\glsnextpages}{}

glossaryentry  If the entrycounter package option has been used, define a counter to number
               each level 0 entry.
3679 \ifglentrycounter
3680   \ifx\@gls@counterwithin\@empty
3681     \newcounter{glossaryentry}
3682   \else
3683     \newcounter{glossaryentry}[\@gls@counterwithin]
3684   \fi
3685   \def\theHglossaryentry{\currentglossary.\theglossaryentry}
3686 \fi

glossarysubentry  If the subentrycounter package option has been used, define a counter to num-
                  ber each level 1 entry.
3687 \ifglssubentrycounter
3688   \ifglentrycounter
3689     \newcounter{glossarysubentry}[glossaryentry]
3690   \else
3691     \newcounter{glossarysubentry}
3692   \fi
3693   \def\theHglossarysubentry{\currentglssubentry.\theglossarysubentry}
3694 \fi

esetsubentrycounter  Resets the glossarysubentry counter.
3695 \ifglssubentrycounter
3696   \newcommand*{\glsresetsubentrycounter}{%
3697     \setcounter{glossarysubentry}{0}%
3698   }
3699 \else
3700   \newcommand*{\glsresetsubentrycounter}{}
3701 \fi

esetsubentrycounter  Resets the glossareentry counter.
3702 \ifglentrycounter
3703   \newcommand*{\glsresetentrycounter}{%
3704     \setcounter{glossaryentry}{0}%
3705   }

```

```

3706 \else
3707   \newcommand*{\glsresetentrycounter}{}
3708 \fi

```

`\glsstepentry` Advance the glossaryentry counter if in use. The argument is the label associated with the entry.

```

3709 \ifglentrycounter
3710   \newcommand*{\glsstepentry}[1]{%
3711     \refstepcounter{glossaryentry}%
3712     \label{glsentry-#1}%
3713   }
3714 \else
3715   \newcommand*{\glsstepentry}[1]{}
3716 \fi

```

`\glsstepsubentry` Advance the glossarysubentry counter if in use. The argument is the label associated with the subentry.

```

3717 \ifglssubentrycounter
3718   \newcommand*{\glsstepsubentry}[1]{%
3719     \def\currentglssubentry{#1}%
3720     \refstepcounter{glossarysubentry}%
3721     \label{glsentry-#1}%
3722   }
3723 \else
3724   \newcommand*{\glsstepsubentry}[1]{}
3725 \fi

```

`\glsrefentry` Reference the entry or sub-entry counter if in use, otherwise just do `\gls`.

```

3726 \ifglentrycounter
3727   \newcommand*{\glsrefentry}[1]{\ref{glsentry-#1}}
3728 \else
3729   \ifglssubentrycounter
3730     \newcommand*{\glsrefentry}[1]{\ref{glsentry-#1}}
3731   \else
3732     \newcommand*{\glsrefentry}[1]{\gls{#1}}
3733   \fi
3734 \fi

```

`glentrycounterlabel` Defines how to display the glossaryentry counter.

```

3735 \ifglentrycounter
3736   \newcommand*{\glentrycounterlabel}{\theglossaryentry.\space}
3737 \else
3738   \newcommand*{\glentrycounterlabel}{}
3739 \fi

```

`glsubentrycounterlabel` Defines how to display the glossarysubentry counter.

```

3740 \ifglssubentrycounter
3741   \newcommand*{\glsubentrycounterlabel}{\theglossarysubentry)\space}

```

```

3742 \else
3743   \newcommand*{\glssubentrycounterlabel}{}
3744 \fi

```

`\glentryitem` Step and display glossaryentry counter, if appropriate.

```

3745 \ifglentrycounter
3746   \newcommand*{\glentryitem}[1]{%
3747     \glstepentry{#1}\glentrycounterlabel
3748   }
3749 \else
3750   \newcommand*{\glentryitem}[1]{\glresetsubentrycounter}
3751 \fi

```

`\glssubentryitem` Step and display glossarysubentry counter, if appropriate.

```

3752 \ifglssubentrycounter
3753   \newcommand*{\glssubentryitem}[1]{%
3754     \glstepsubentry{#1}\glssubentrycounterlabel
3755   }
3756 \else
3757   \newcommand*{\glssubentryitem}[1]{}
3758 \fi

```

`theglossary` If the `theglossary` environment has already been defined, a warning will be issued. This environment should be redefined by glossary styles.

```

3759 \ifcsundef{theglossary}%
3760 {%
3761   \newenvironment{theglossary}{}{}%
3762 }%
3763 {%
3764   \GlossariesWarning{overriding ‘theglossary’ environment}%
3765   \renewenvironment{theglossary}{}{}%
3766 }

```

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`

```

3767 \newcommand*{\glossaryheader}{}

```

`\glstarget` `\glstarget{<label>}{<name>}`

Provide user interface to `\@glstarget` to make it easier to modify the glossary style in the document.

```

3768 \newcommand*{\glstarget}[2]{\@glstarget{\glolinkprefix#1}{#2}}

```

`\glossaryentryfield` `\glossaryentryfield{<label>}{<name>}{<description>}{<symbol>}{<page-list>}`

This command governs how each entry row should be formatted in the glossary. Glossary styles need to redefine this command. Most of the predefined styles ignore `<symbol>`.

```
3769 \newcommand*{\glossaryentryfield}[5]{%
3770 \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.)

```
3771 \newcommand*{\glossarysubentryfield}[6]{%
3772 \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`

```
3773 \newcommand*{\glsgroupskip}{}
```

Each of the 28 glossary groups described above is preceded by a group heading. This is formatted by the command `\glsgroupheading` which takes one argument which is the *label* assigned to that group (not the title). The corresponding labels are: `glssymbols`, `glsnumbers`, A, ..., Z. Glossary styles must redefine this command. (In between groups, `\glsgroupheading` comes immediately after `\glsgroupskip`.)

`\glsgroupheading`

```
3774 \newcommand*{\glsgroupheading}[1]{}
```

It is possible to “trick” `makeindex` into treating entries as though they belong to the same group, even if the terms don’t start with the same letter, by modifying the sort key. For example, all entries belonging to one group could be defined so that the sort key starts with an a, while entries belonging to another group could be defined so that the sort key starts with a b, and so on. If you want each group to have a heading, you would then need to modify the translation control sequences `\glsgrouptitle` and `\glsgrouplabel` so that the label

is translated into the required title (and vice-versa).

`\glsgetgrouptitle{<label>}`

This command produces the title for the glossary group whose label is given by `<label>`. By default, the group labelled `glssymbols` produces `\glssymbolsgroupname`, the group labelled `glsnumbers` produces `\glsnumbersgroupname` and all the other groups simply produce their label. As mentioned above, the group labels are: `glssymbols`, `glsnumbers`, `A`, ..., `Z`. If you want to redefine the group titles, you will need to redefine this command.

`\glsgetgrouptitle`

```
3775 \newcommand*{\glsgetgrouptitle}[1]{%
3776   \ifcsundef{#1groupname}{#1}{\csname #1groupname\endcsname}%
3777 }
```

`\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`

```
3778 \newcommand*{\glsgetgrouplabel}[1]{%
3779 \ifthenelse{\equal{#1}{\glssymbolsgroupname}}{\glssymbols}{%
3780 \ifthenelse{\equal{#1}{\glsnumbersgroupname}}{\glsnumbers}{#1}}}
```

The command `\setentrycounter` sets the entry's associated counter (required by `\glshypernumber` etc.) `\glslink` and `\glsadd` encode the `\glossary` argument so that the relevant counter is set prior to the formatting command.

`\setentrycounter`

```
3781 \newcommand*{\setentrycounter}[2][ ]{%
3782   \def\@glo@counterprefix{#1}%
3783   \ifx\@glo@counterprefix\@empty
3784     \def\@glo@counterprefix{.}%
3785   \else
3786     \def\@glo@counterprefix{.#1}%
3787   \fi
3788   \def\glsentrycounter{#2}%
3789 }
```

The current glossary style can be set using `\glossarystyle{<style>}`.

`\glossarystyle`

```
3790 \newcommand*{\glossarystyle}[1]{%
3791   \ifcsundef{@glsstyle@#1}%
3792   {%
3793     \PackageError{glossaries}{Glossary style ‘#1’ undefined}{%}
```

```

3794 }%
3795 {%
3796     \csname @glsstyle@#1\endcsname
3797 }%
3798 }

```

`\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.

```

3799 \newcommand{\newglossarystyle}[2]{%
3800     \ifcsundef{@glsstyle@#1}%
3801     {%
3802         \expandafter\def\csname @glsstyle@#1\endcsname{#2}%
3803     }%
3804     {%
3805         \PackageError{glossaries}{Glossary style ‘#1’ is already defined}{}%
3806     }%
3807 }

```

`\renewglossarystyle` Code for this macro supplied by Marco Daniel.

```

3808 \newcommand{\renewglossarystyle}[2]{%
3809     \ifcsundef{@glsstyle@#1}%
3810     {%
3811         \PackageError{glossaries}{Glossary style ‘#1’ isn’t already defined}{}%
3812     }%
3813     {%
3814         \csdef{@glsstyle@#1}{#2}%
3815     }%
3816 }

```

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`

```
3817 \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`

```
3818 \ifcsundef{hyperlink}%
3819 {%
3820   \def\glshypernumber#1{#1}%
3821 }%
3822 {%
3823   \def\glshypernumber#1{\@glshypernumber#1\nohyperpage{}}\@nil}
3824 }
```

`\@glshypernumber` This code was provided by Heiko Oberdiek to allow material to be attached to the location.

```
3825 \def\@glshypernumber#1\nohyperpage#2#3\@nil{%
3826   \ifx\#1\%
3827     \else
3828       \@delimR#1\delimR\delimR\%
3829     \fi
3830     \ifx\#2\%
3831       \else
3832         #2%
3833       \fi
3834       \ifx\#3\%
3835         \else
3836           \@glshypernumber#3\@nil
3837         \fi
3838 }
```

`\@delimR` displays a range of numbers for the counter whose name is given by `\@gls@counter` (which must be set prior to using `\glshypernumber`).

`\@delimR`

```
3839 \def\@delimR#1\delimR #2\delimR #3\{%
3840 \ifx\#2\%
3841   \@delimN{#1}%
3842 \else
3843   \@gls@numberlink{#1}\delimR\@gls@numberlink{#2}%
3844 \fi}
```

`\@delimN` displays a list of individual numbers, instead of a range:

`\@delimN`

```

3845 \def\@delimN#1{\@delimN#1\delimN \delimN\}
3846 \def\@delimN#1\delimN #2\delimN#3\{\{
3847 \ifx\#3\}%
3848   \@gls@numberlink{#1}%
3849 \else
3850   \@gls@numberlink{#1}\delimN\@gls@numberlink{#2}%
3851 \fi
3852 }
```

The following code is modified from hyperref's `\HyInd@pagelink` where the name of the counter being used is given by `\@gls@counter`.

```

3853 \def\@gls@numberlink#1{%
3854 \begingroup
3855   \toks@={}%
3856   \@gls@removespaces#1 \@nil
3857 \endgroup}

3858 \def\@gls@removespaces#1 #2\@nil{%
3859   \toks@=\expandafter{\the\toks@#1}%
3860   \ifx\#2\}%
3861     \edef\x{\the\toks@}%
3862     \ifx\x\empty
3863       \else

3864         \hyperlink{\glsentrycounter\@glo@counterprefix\the\toks@}%
3865           {\the\toks@}%
3866       \fi
3867     \else
3868       \@gls@ReturnAfterFi{%
3869         \@gls@removespaces#2\@nil
3870       }%
3871     \fi
3872 }
3873 \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`

```

3874 \newcommand*\hyperrm[1]{\textrm{\glshypernumber{#1}}}
```

`\hypersf`

```

3875 \newcommand*\hypersf[1]{\textsf{\glshypernumber{#1}}}
```

`\hypertt`

```

3876 \newcommand*\hypertt[1]{\texttt{\glshypernumber{#1}}}
```

```

\hyperbf
3877 \newcommand*{\hyperbf}[1]{\textbf{\glshypernumber{#1}}}

\hypermd
3878 \newcommand*{\hypermd}[1]{\textmd{\glshypernumber{#1}}}

\hyperit
3879 \newcommand*{\hyperit}[1]{\textit{\glshypernumber{#1}}}

\hypersl
3880 \newcommand*{\hypersl}[1]{\textsl{\glshypernumber{#1}}}

\hyperup
3881 \newcommand*{\hyperup}[1]{\textup{\glshypernumber{#1}}}

\hypersc
3882 \newcommand*{\hypersc}[1]{\textsc{\glshypernumber{#1}}}

\hyperemph
3883 \newcommand*{\hyperemph}[1]{\emph{\glshypernumber{#1}}}

```

## 1.16 Acronyms

If the acronym package option is used, a new glossary called acronym is created

```

3884 \ifglsacronym
3885 \newglossary[alg]{acronym}{acr}{acn}{\acronymname}
    and \acronymtype is set to the name of this new glossary.
3886 \renewcommand*{\acronymtype}{acronym}
3887 \fi

```

```

\oldacronym \oldacronym[\langle label \rangle]{\langle abbrev \rangle}{\langle long \rangle}{\langle key-val list \rangle}

```

This emulates the way the old package defined acronyms. It is equivalent to `\newacronym[\langle key-val list \rangle]{\langle label \rangle}{\langle abbrev \rangle}{\langle long \rangle}` and it additionally defines the command `\langle label \rangle` which is equivalent to `\gls{\langle label \rangle}` (thus `\langle label \rangle` must only contain alphabetical characters). If `\langle label \rangle` is omitted, `\langle abbrev \rangle` is used. This only emulates the syntax of the old package. The way the acronyms appear in the list of acronyms is determined by the definition of `\newacronym` and the glossary style.

Note that `\langle label \rangle` can't have an optional argument if the package is loaded. If hasn't been loaded then you can do `\langle label \rangle[\langle insert \rangle]` but you can't do `\langle label \rangle[\langle key-val list \rangle]`. For example if you define the acronym `svm`, then you can do `\svm['s]` but you can't do `\svm[format=textbf]`. If the package is loaded, `\svm['s]` will appear as `svm ['s]` which is unlikely to be the desired

result. In this case, you will need to use `\gls` explicitly, e.g. `\gls{svm}[’s]`.  
 Note that it is up to the user to load if desired.

```

3888 \newcommand{\oldacronym}[4][\gls@label]{%
3889   \def\gls@label{#2}%
3890   \newacronym[#4]{#1}{#2}{#3}%
3891   \ifcsundef{xspace}%
3892   {%
3893     \expandafter\edef\csname#1\endcsname{%
3894       \noexpand\@ifstar{\noexpand\Gls{#1}}{\noexpand\gls{#1}}%
3895     }%
3896   }%
3897   {%
3898     \expandafter\edef\csname#1\endcsname{%
3899       \noexpand\@ifstar{\noexpand\Gls{#1}\noexpand\xspace}{%
3900         \noexpand\gls{#1}\noexpand\xspace}%
3901     }%
3902   }%
3903 }
```

`\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`

```

3904 \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.

```

3905 \newcommand*{\acrpluralsuffix}{\glspluralsuffix}
```

The following are defined for compatibility with version 2.07 and earlier.

`\glsshortkey`

```

3906 \newcommand*{\glsshortkey}{short}
```

```

\glsshortpluralkey
3907 \newcommand*{\glsshortpluralkey}{shortplural}

\glslongkey
3908 \newcommand*{\glslongkey}{long}

\glslongpluralkey
3909 \newcommand*{\glslongpluralkey}{longplural}

\acrfull  Full form of the acronym.
3910 \newrobustcmd*{\acrfull}{%
3911   \@ifstar\s@acrfull\ns@acrfull
3912 }

3913 \newcommand*\s@acrfull[2][{}]{%
3914   \new@ifnextchar[{\@acrfull{hyper=false,#1}{#2}}}%
3915   {\@acrfull{hyper=false,#1}{#2}[]}%
3916 }
3917 \newcommand*\ns@acrfull[2][{}]{%
3918   \new@ifnextchar[{\@acrfull{#1}{#2}}}%
3919   {\@acrfull{#1}{#2}[]}%
3920 }

    Low-level macro:
3921 \def\@acrfull#1#2[#3]{%
3922   \acrlinkfullformat{\@acrlong}{\@acrshort}{#1}{#2}{#3}%
3923 }

\acrlinkfullformat  Format for full links like \acrfull.  Syntax: \acrlinkfullformat{<long
cs>}{<short cs>}{<options>}{<label>}{<insert>}
3924 \newcommand{\acrlinkfullformat}[5]{%
3925   \acrfullformat{#1{#3}{#4}[#5]}{#2{#3}{#4}[]}%
3926 }

\acrfullformat  Default full form is <long> (<short>).
3927 \newcommand{\acrfullformat}[2]{#1\space(#2)}

    Default format for full acronym

\Acrfull
3928 \newrobustcmd*{\Acrfull}{%
3929   \@ifstar\s@Acrfull\ns@Acrfull
3930 }

3931 \newcommand*\s@Acrfull[2][{}]{%
3932   \new@ifnextchar[{\@Acrfull{hyper=false,#1}{#2}}}%
3933   {\@Acrfull{hyper=false,#1}{#2}[]}%
3934 }
3935 \newcommand*\ns@Acrfull[2][{}]{%

```

```

3936 \new@ifnextchar[{\@Acrfull{#1}{#2}}%
3937         {\@Acrfull{#1}{#2} []}%
3938 }

```

Low-level macro:

```

3939 \def\@Acrfull#1#2[#3]{%
3940 \acrlinkfullformat{\@Acrlong}{\@acrshort}{#1}{#2}{#3}%
3941 }

```

\ACRfull

```

3942 \newrobustcmd*{\ACRfull}{%
3943 \@ifstar\s@ACRfull\ns@ACRfull
3944 }

3945 \newcommand*\s@ACRfull[2] []{%
3946 \new@ifnextchar[{\@ACRfull{hyper=false,#1}{#2}}%
3947         {\@ACRfull{hyper=false,#1}{#2} []}%
3948 }

3949 \newcommand*\ns@ACRfull[2] []{%
3950 \new@ifnextchar[{\@ACRfull{#1}{#2}}%
3951         {\@ACRfull{#1}{#2} []}%
3952 }

```

Low-level macro:

```

3953 \def\@ACRfull#1#2[#3]{%
3954 \acrlinkfullformat{\@ACRlong}{\@ACRshort}{#1}{#2}{#3}%
3955 }

```

Plural:

\acrfullpl

```

3956 \newrobustcmd*{\acrfullpl}{%
3957 \@ifstar\s@acrfullpl\ns@acrfullpl
3958 }

3959 \newcommand*\s@acrfullpl[2] []{%
3960 \new@ifnextchar[{\@acrfullpl{hyper=false,#1}{#2}}%
3961         {\@acrfullpl{hyper=false,#1}{#2} []}%
3962 }

3963 \newcommand*\ns@acrfullpl[2] []{%
3964 \new@ifnextchar[{\@acrfullpl{#1}{#2}}%
3965         {\@acrfullpl{#1}{#2} []}%
3966 }

```

Low-level macro:

```

3967 \def\@acrfullpl#1#2[#3]{%
3968 \acrlinkfullformat{\@acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%
3969 }

```

`\Acrfullpl`

```
3970 \newrobustcmd*{\Acrfullpl}{%
3971   \@ifstar\s@Acrfullpl\ns@Acrfullpl
3972 }

3973 \newcommand*\s@Acrfullpl[2][]{%
3974   \new@ifnextchar[{\@Acrfullpl{hyper=false,#1}{#2}}%
3975     {\@Acrfullpl{hyper=false,#1}{#2}[]}%
3976 }
3977 \newcommand*\ns@Acrfullpl[2][]{%
3978   \new@ifnextchar[{\@Acrfullpl{#1}{#2}}%
3979     {\@Acrfullpl{#1}{#2}[]}%
3980 }
```

Low-level macro:

```
3981 \def\@Acrfullpl#1#2[#3]{%
3982   \acrlinkfullformat{\@Acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%
3983 }
```

`\ACRfullpl`

```
3984 \newrobustcmd*{\ACRfullpl}{%
3985   \@ifstar\s@ACRfullpl\ns@ACRfullpl
3986 }

3987 \newcommand*\s@ACRfullpl[2][]{%
3988   \new@ifnextchar[{\@ACRfullpl{hyper=false,#1}{#2}}%
3989     {\@ACRfullpl{hyper=false,#1}{#2}[]}%
3990 }
3991 \newcommand*\ns@ACRfullpl[2][]{%
3992   \new@ifnextchar[{\@ACRfullpl{#1}{#2}}%
3993     {\@ACRfullpl{#1}{#2}[]}%
3994 }
```

Low-level macro:

```
3995 \def\@ACRfullpl#1#2[#3]{%
3996   \acrlinkfullformat{\@ACRlongpl}{\@ACRshortpl}{#1}{#2}{#3}%
3997 }
```

## 1.17 Predefined acronym styles

`\acronymfont` This is only used with the additional acronym styles:

```
3998 \newcommand{\acronymfont}[1]{#1}
```

`\firstacronymfont` This is only used with the additional acronym styles:

```
3999 \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.

```
4000 \newcommand*\acrnameformat[2]{\acronymfont{#1}}
```

Define some tokens used by \newacronym:

\glskeylisttok

4001 \newtoks\glskeylisttok

\glslabeltok

4002 \newtoks\glslabeltok

\glsshorttok

4003 \newtoks\glsshorttok

\glslongtok

4004 \newtoks\glslongtok

\newacronymhook Provide a hook for \newacronym:

4005 \newcommand\*{\newacronymhook}{}

AcronymDisplayStyle Sets the default acronym display style for given glossary.

```
4006 \newcommand*{\SetDefaultAcronymDisplayStyle}[1]{%
4007   \defglsdisplay[#1]{##1##4}%
4008   \defglsdisplayfirst[#1]{##1##4}%
4009 }
```

DefaultNewAcronymDef Sets up the acronym definition for the default style. The information is provided by the tokens \glslabeltok, \glsshorttok, \glslongtok and \glskeylisttok.

```
4010 \newcommand*{\DefaultNewAcronymDef}{%
4011   \edef\@do@newglossaryentry{%
4012     \noexpand\newglossaryentry{\the\glslabeltok}%
4013     {%
4014       type=\acronymtype,%
4015       name={\the\glsshorttok},%
4016       sort={\the\glsshorttok},%
4017       text={\the\glsshorttok},%
4018       first={\acrfullformat{\the\glslongtok}{\the\glsshorttok}},%
4019       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4020       firstplural={\acrfullformat{\noexpand\@glo@longpl}%
4021         {\noexpand\@glo@shortpl}},%
4022       short={\the\glsshorttok},%
4023       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4024       long={\the\glslongtok},%
4025       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4026       description={\the\glslongtok},%
4027       descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
```

Remaining options specified by the user:

```
4028     \the\glskeylisttok
4029   }%
```

```

4030 }%
4031 \@do@newglossaryentry
4032 }

```

**DefaultAcronymStyle** Set up the default acronym style:

```

4033 \newcommand*\SetDefaultAcronymStyle{%
    Set the display style:
4034 \@for\@gls@type:=\@glsacronymlists\do{%
4035     \SetDefaultAcronymDisplayStyle{\@gls@type}%
4036 }%

```

Set up the definition of \newacronym:

```

4037 \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).
4038     \ifx\@glsacronymlists\@empty
4039         \def\@glo@type{\acronymtype}%
4040         \setkeys{glossentry}{##1}%
4041         \DeclareAcronymList{\@glo@type}%
4042         \SetDefaultAcronymDisplayStyle{\@glo@type}%
4043     \fi
4044     \glskeylisttok{##1}%
4045     \glslabeltok{##2}%
4046     \glsshorttok{##3}%
4047     \glslongtok{##4}%
4048     \newacronymhook
4049     \DefaultNewAcronymDef
4050 }%
4051 \renewcommand*\acrpluralsuffix{\glspluralsuffix}%
4052 }

```

**\acrfootnote** Used by the footnote acronym styles.

```

4053 \newcommand*\acrfootnote[3]{\acrlinkfootnote{#1}{#2}{#3}}

```

**\acrlinkfootnote**

```

4054 \newcommand*\acrlinkfootnote[3]{%
4055     \footnote{\glslink[#1]{#2}{#3}}%
4056 }

```

**\acrlinkfootnote**

```

4057 \newcommand*\acrlinkfootnote[3]{%
4058     \footnote{#3}%
4059 }

```

**AcronymDisplayStyle** Sets the acronym display style for given glossary for the description and footnote combination.

```

4060 \newcommand*\SetDescriptionFootnoteAcronymDisplayStyle}[1]{%
4061   \defglsdisplayfirst[#1]{%
4062     \firstacronymfont{##1}##4%
4063     \expandafter\protect\expandafter\acrfootnote\expandafter
4064       {\@gls@link@opts}{\@gls@link@label}{##3}%
4065   }%
4066   \defglsdisplay[#1]{\acronymfont{##1}##4}%
4067 }

```

otnoteNewAcronymDef

```

4068 \newcommand*\DescriptionFootnoteNewAcronymDef{%
4069   \edef\@do@newglossaryentry{%
4070     \noexpand\newglossaryentry{\the\glslabeltok}%
4071     {%
4072       type=\acronymtype,%
4073       name={\noexpand\acronymfont{\the\glsshorttok}},%
4074       sort={\the\glsshorttok},%
4075       text={\the\glsshorttok},%
4076       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4077       short={\the\glsshorttok},%
4078       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4079       long={\the\glslongtok},%
4080       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4081       symbol={\the\glslongtok},%
4082       symbolplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4083       \the\glskeylisttok
4084     }%
4085   }%
4086   \@do@newglossaryentry
4087 }

```

ootnoteAcronymStyle If a description and footnote are both required, store the long form in the symbol key. Store the short form in text key. Note that since the long form is stored in the symbol key, if you want the long form to appear in the list of acronyms, you need to use a glossary style that displays the symbol key.

```

4088 \newcommand*\SetDescriptionFootnoteAcronymStyle{%
4089   \renewcommand{\newacronym}[4][\]{%
4090     \ifx\@glsacronymlists\@empty
4091       \def\@glo@type{\acronymtype}%
4092       \setkeys{glossentry}{##1}%
4093       \DeclareAcronymList{\@glo@type}%
4094       \SetDescriptionFootnoteAcronymDisplayStyle{\@glo@type}%
4095     \fi
4096     \glskeylisttok{##1}%
4097     \glslabeltok{##2}%
4098     \glsshorttok{##3}%
4099     \glslongtok{##4}%
4100     \newacronymhook
4101     \DescriptionFootnoteNewAcronymDef

```

4102 }%

If footnote package option is specified, set the first use to append the long form (stored in symbol) as a footnote.

```
4103 \@for\@gls@type:=\@glsacronymlists\do{%
4104   \SetDescriptionFootnoteAcronymDisplayStyle{\@gls@type}%
4105 }%
```

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.

```
4106 \ifglsacrsmallcaps
4107   \renewcommand*{\acronymfont}[1]{\textsc{##1}}%
4108   \renewcommand*{\acrpluralsuffix}{%
4109     \textup{\glspluralsuffix}}%
4110 \else
4111   \ifglsacrsmaller
4112     \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%
4113   \fi
4114 \fi
```

Check for package option clash

```
4115 \ifglsacrdua
4116   \PackageError{glossaries}{Option clash: ‘footnote’ and ‘dua’
4117     can’t both be set}{}%
4118 \fi
4119 }%
```

**AcronymDisplayStyle** Sets the acronym display style for given glossary with description and dua combination.

```
4120 \newcommand*{\SetDescriptionDUAAcronymDisplayStyle}[1]{%
4121   \defglsdisplay[#1]{##1##4}%
4122   \defglsdisplayfirst[#1]{##1##4}%
4123 }
```

**ionDUANewAcronymDef**

```
4124 \newcommand*{\DescriptionDUANewAcronymDef}{%
4125   \edef\@do@newglossaryentry{%
4126     \noexpand\newglossaryentry{\the\glslabeltok}%
4127     {%
4128       type=\acronymtype,%
4129       name={\the\glslongtok},%
4130       sort={\the\glslongtok},%
4131       text={\the\glslongtok},%
4132       plural={\the\glslongtok\noexpand\acrpluralsuffix},%
4133       short={\the\glsshorttok},%
4134       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4135       long={\the\glslongtok},%
4136       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4137       symbol={\the\glsshorttok},%
```

```

4138     symbolplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4139     \the\glskeylisttok
4140   }%
4141 }%
4142 \@do@newglossaryentry
4143 }

```

**tionDUAAcronymStyle** Description, don't use acronym and no footnote. Note that the short form is stored in the symbol key, so if the short form needs to be displayed in the glossary, use a style the displays the symbol.

```

4144 \newcommand*{\SetDescriptionDUAAcronymStyle}{%
4145   \ifglssacrsmallcaps
4146     \PackageError{glossaries}{Option clash: 'smallcaps' and 'dua'
4147       can't both be set}{}%
4148   \else
4149     \ifglssacrsmaller
4150       \PackageError{glossaries}{Option clash: 'smaller' and 'dua'
4151         can't both be set}{}%
4152     \fi
4153   \fi
4154   \renewcommand{\newacronym}[4][[]]{%
4155     \ifx\@glsacronymlists\@empty
4156       \def\@glo@type{\acronymtype}%
4157       \setkeys{glossentry}{##1}%
4158       \DeclareAcronymList{\@glo@type}%
4159       \SetDescriptionDUAAcronymDisplayStyle{\@glo@type}%
4160     \fi
4161     \glskeylisttok{##1}%
4162     \glslabeltok{##2}%
4163     \glsshorttok{##3}%
4164     \glslongtok{##4}%
4165     \newacronymhook
4166     \DescriptionDUANewAcronymDef
4167   }%

```

Set display.

```

4168   \@for\@gls@type:=\@glsacronymlists\do{%
4169     \SetDescriptionDUAAcronymDisplayStyle{\@gls@type}%
4170   }%
4171 }%

```

**AcronymDisplayStyle** Sets the acronym display style for given glossary using the description setting (but not footnote or dua).

```

4172 \newcommand*{\SetDescriptionAcronymDisplayStyle}[1]{%
4173   \defglssdisplayfirst[#1]{%
4174     ##1##4 (\firstacronymfont{##3})}%
4175   \defglssdisplay[#1]{\acronymfont{##1}##4}%
4176 }

```

ptionNewAcronymDef

```

4177 \newcommand*{\DescriptionNewAcronymDef}{%
4178   \edef\@do@newglossaryentry{%
4179     \noexpand\newglossaryentry{\the\glslabeltok}%
4180     {%
4181       type=\acronymtype,%
4182       name={\noexpand
4183         \acrnameformat{\the\glsshorttok}{\the\glslongtok}},%
4184       sort={\the\glsshorttok},%
4185       first={\the\glslongtok},%
4186       firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4187       text={\the\glsshorttok},%
4188       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4189       short={\the\glsshorttok},%
4190       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4191       long={\the\glslongtok},%
4192       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4193       symbol={\noexpand\@glo@text},%
4194       symbolplural={\noexpand\@glo@plural},%
4195       \the\glskeylisttok}%
4196   }%
4197   \@do@newglossaryentry
4198 }

```

riptionAcronymStyle

Option description is used, but not dua or footnote. Store long form in first key and short form in text and symbol key. The name is stored using `\acrnameformat` to allow the user to override the way the name is displayed in the list of acronyms.

```

4199 \newcommand*{\SetDescriptionAcronymStyle}{%
4200   \renewcommand{\newacronym}[4][\]{%
4201     \ifx\@glsacronymlists\@empty
4202       \def\@glo@type{\acronymtype}%
4203       \setkeys{glossentry}{##1}%
4204       \DeclareAcronymList{\@glo@type}%
4205       \SetDescriptionAcronymDisplayStyle{\@glo@type}%
4206     \fi
4207     \glskeylisttok{##1}%
4208     \glslabeltok{##2}%
4209     \glsshorttok{##3}%
4210     \glslongtok{##4}%
4211     \newacronymhook
4212     \DescriptionNewAcronymDef
4213   }%

```

Set display.

```

4214   \@for\@gls@type:=\@glsacronymlists\do{%
4215     \SetDescriptionAcronymDisplayStyle{\@gls@type}%
4216   }%

```

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.

```

4217 \ifglsmallcaps
4218   \renewcommand{\acronymfont}[1]{\textsc{##1}}
4219   \renewcommand*{\acrpluralsuffix}{%
4220     \textup{\glpluralsuffix}}%
4221 \else
4222   \ifglsmaller
4223     \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}%
4224   \fi
4225 \fi
4226 }%
```

**AcronymDisplayStyle** Sets the acronym display style for given glossary with footnote setting (but not description or dua).

```

4227 \newcommand*\SetFootnoteAcronymDisplayStyle[1]{%
4228   \defgldisplayfirst[#1]{%
4229     \firstacronymfont{##1}##4%
4230     \expandafter\protect\expandafter\acrfootnote\expandafter
4231       {\@gls@link@opts}{\@gls@link@label}{##2}%
4232   }%
4233   \defgldisplay[#1]{\acronymfont{##1}##4}%
4234 }
```

**FootnoteNewAcronymDef**

```

4235 \newcommand*\FootnoteNewAcronymDef{%
4236   \edef\@do@newglossaryentry{%
4237     \noexpand\newglossaryentry{\the\glslabeltok}%
4238     {%
4239       type=\acronymtype,%
4240       name={\noexpand\acronymfont{\the\glsshorttok}},%
4241       sort={\the\glsshorttok},%
4242       text={\the\glsshorttok},%
4243       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4244       short={\the\glsshorttok},%
4245       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4246       long={\the\glslongtok},%
4247       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4248       description={\the\glslongtok},%
4249       descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4250       \the\glskeylisttok
4251     }%
4252   }%
4253   \@do@newglossaryentry
4254 }
```

**FootnoteAcronymStyle** If footnote package option is specified, set the first use to append the long form

(stored in description) as a footnote. Use the description key to store the long form.

```

4255 \newcommand*\SetFootnoteAcronymStyle{%
4256   \renewcommand{\newacronym}[4][\]{%
4257     \ifx\@glsacronymlists\@empty
4258       \def\@glo@type{\acronymtype}%
4259       \setkeys{glossentry}{##1}%
4260       \DeclareAcronymList{\@glo@type}%
4261       \SetFootnoteAcronymDisplayStyle{\@glo@type}%
4262     \fi
4263     \glskeylisttok{##1}%
4264     \glslabeltok{##2}%
4265     \glsshorttok{##3}%
4266     \glslongtok{##4}%
4267     \newacronymhook
4268     \FootnoteNewAcronymDef
4269   }%

```

Set display

```

4270   \@for\@gls@type:=\@glsacronymlists\do{%
4271     \SetFootnoteAcronymDisplayStyle{\@gls@type}%
4272   }%

```

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.

```

4273   \ifglsacrsmallcaps
4274     \renewcommand*\acronymfont[1]{\textsc{##1}}%
4275     \renewcommand*\acrpluralsuffix{%
4276       \textup{\glspluralsuffix}}%
4277   \else
4278     \ifglsacrsmaller
4279       \renewcommand*\acronymfont[1]{\textsmaller{##1}}%
4280     \fi
4281   \fi

```

Check for option clash

```

4282   \ifglsacrdua
4283     \PackageError{glossaries}{Option clash: ‘footnote’ and ‘dua’
4284       can’t both be set}{}%
4285   \fi
4286 }%

```

**AcronymDisplayStyle** Sets the acronym display style for given glossary where neither footnote nor description is required, but smallcaps or smaller specified.

```

4287 \newcommand*\SetSmallAcronymDisplayStyle[1]{%
4288   \defglsdisplayfirst[#1]{##1##4 (\firstacronymfont{##3})}
4289   \defglsdisplay[#1]{\acronymfont{##1}##4}%
4290 }

```

\SmallNewAcronymDef

```

4291 \newcommand*{\SmallNewAcronymDef}{%
4292   \edef\@do@newglossaryentry{%
4293     \noexpand\newglossaryentry{\the\glslabeltok}%
4294     {%
4295       type=\acronymtype,%
4296       name={\noexpand\acronymfont{\the\glsshorttok}},%
4297       sort={\the\glsshorttok},%
4298       text={\noexpand\@glo@symbol},%

```

Default to the short plural.

```

4299     plural={\noexpand\@glo@shortpl},%
4300     first={\the\glslongtok},%

```

Default to the long plural.

```

4301     firstplural={\noexpand\@glo@longpl},%
4302     short={\the\glsshorttok},%
4303     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4304     long={\the\glslongtok},%
4305     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4306     description={\noexpand\@glo@first},%
4307     descriptionplural={\noexpand\@glo@firstplural},%
4308     symbol={\the\glsshorttok},%

```

Default to the short plural.

```

4309     symbolplural={\noexpand\@glo@shortpl},%
4310     \the\glskeylisttok
4311   }%
4312 }%
4313 \@do@newglossaryentry
4314 }

```

etSmallAcronymStyle Neither footnote nor description required, but smallcaps or smaller specified.

Use the symbol key to store the short form and first to store the long form.

```

4315 \newcommand*{\SetSmallAcronymStyle}{%
4316   \renewcommand{\newacronym}[4][\]{%
4317     \ifx\@glsacronymlists\@empty
4318       \def\@glo@type{\acronymtype}%
4319       \setkeys{glossentry}{##1}%
4320       \DeclareAcronymList{\@glo@type}%
4321       \SetSmallAcronymDisplayStyle{\@glo@type}%
4322     \fi
4323     \glskeylisttok{##1}%
4324     \glslabeltok{##2}%
4325     \glsshorttok{##3}%
4326     \glslongtok{##4}%
4327     \newacronymhook
4328     \SmallNewAcronymDef
4329   }%

```

Change the display since first only contains long form.

```
4330 \@for\@gls@type:=\@glsacronymlists\do{%
4331     \SetSmallAcronymDisplayStyle{\@gls@type}%
4332 }%
```

Redefine \acronymfont if small caps required. The plural suffix is set in an up-right font so that it remains in normal lower case, otherwise it looks as though it's part of the acronym.

```
4333 \ifglsacrsmallcaps
4334     \renewcommand*{\acronymfont}[1]{\textsc{##1}}
4335     \renewcommand*{\acrpluralsuffix}{%
4336         \textup{\glspluralsuffix}}%
4337 \else
4338     \renewcommand*{\acronymfont}[1]{\textsmaller{##1}}
4339 \fi
```

check for option clash

```
4340 \ifglsacrdua
4341     \ifglsacrsmallcaps
4342         \PackageError{glossaries}{Option clash: ‘smallcaps’ and ‘dua’
4343             can’t both be set}{}%
4344     \else
4345         \PackageError{glossaries}{Option clash: ‘smaller’ and ‘dua’
4346             can’t both be set}{}%
4347     \fi
4348 \fi
4349 }%
```

`\SetDUADisplayStyle` Sets the acronym display style for given glossary with dua setting.

```
4350 \newcommand*{\SetDUADisplayStyle}[1]{%
4351     \defglsdisplay[##1]{##1##4}%
4352     \defglsdisplayfirst[##1]{##1##4}%
4353 }
```

`\DUANewAcronymDef`

```
4354 \newcommand*{\DUANewAcronymDef}{%
4355     \edef\@do@newglossaryentry{%
4356         \noexpand\newglossaryentry{\the\glslabeltok}%
4357         {%
4358             type=\acronymtype,%
4359             name={\the\glsshorttok},%
4360             text={\the\glslongtok},%
4361             plural={\the\glslongtok\noexpand\acrpluralsuffix},%
4362             short={\the\glsshorttok},%
4363             shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4364             long={\the\glslongtok},%
4365             longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4366             description={\the\glslongtok},%
4367             symbol={\the\glsshorttok},%
```

```

4368     symbolplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4369     \the\glskeylisttok
4370   }%
4371 }%
4372 \do@newglossaryentry
4373 }

```

`\SetDUASStyle` Always expand acronyms.

```

4374 \newcommand*{\SetDUASStyle}{%
4375   \renewcommand{\newacronym}[4][\{%
4376     \ifx\@glsacronymlists\@empty
4377       \def\@glo@type{\acronymtype}%
4378       \setkeys{glossentry}{##1}%
4379       \DeclareAcronymList{\@glo@type}%
4380       \SetDUADisplayStyle{\@glo@type}%
4381     \fi
4382     \glskeylisttok{##1}%
4383     \glslabeltok{##2}%
4384     \glsshorttok{##3}%
4385     \glslongtok{##4}%
4386     \newacronymhook
4387     \DUANewAcronymDef
4388   }%

```

Set the display

```

4389   \@for\@gls@type:=\@glsacronymlists\do{%
4390     \SetDUADisplayStyle{\@gls@type}%
4391   }%
4392 }

```

`\SetAcronymStyle`

```

4393 \newcommand*{\SetAcronymStyle}{%
4394   \SetDefaultAcronymStyle
4395   \ifglsacrdescription
4396     \ifglsacrfootnote
4397       \SetDescriptionFootnoteAcronymStyle
4398     \else
4399       \ifglsacrdua
4400         \SetDescriptionDUAAcronymStyle
4401       \else
4402         \SetDescriptionAcronymStyle
4403       \fi
4404     \fi
4405   \else
4406     \ifglsacrfootnote
4407       \SetFootnoteAcronymStyle
4408     \else
4409       \ifthenelse{\boolean{glsacrsmalldcaps}}\OR
4410         \boolean{glsacrsmaller}}}%
4411   {%

```

```

4412         \SetSmallAcronymStyle
4413     }%
4414     {%
4415         \ifglssacrdua
4416         \SetDUASstyle
4417         \fi
4418     }%
4419     \fi
4420     \fi
4421 }

```

Set the acronym style according to the package options

```

4422 \SetAcronymStyle

```

Allow user to define their own custom acronyms. (For compatibility with versions before v3.0, the short form is stored in the user1 key, the plural short form is stored in the user2 key, the long form is stored in the user3 key and the plural long form is stored in the user4 key.) Defaults to displaying only the acronym with the long form as the description.

**CustomDisplayStyle** Sets the acronym display style.

```

4423 \newcommand*\SetCustomDisplayStyle}[1]{%
4424     \defglssdisplay[#1]{##1##4}%
4425     \defglssdisplayfirst[#1]{##1##4}%
4426 }

```

**CustomAcronymFields**

```

4427 \newcommand*\CustomAcronymFields{%
4428     name={\the\glssshorttok},%
4429     description={\the\glslongtok},%
4430     first={\noexpand\acrfullformat{\the\glslongtok}{\the\glssshorttok}},%
4431     firstplural={\noexpand\acrfullformat
4432         {\the\glslongtok\noexpand\acrpluralsuffix}{\the\glssshorttok}}%
4433     text={\the\glssshorttok},%
4434     plural={\the\glssshorttok\noexpand\acrpluralsuffix}%
4435 }

```

**CustomNewAcronymDef**

```

4436 \newcommand*\CustomNewAcronymDef{%
4437     \protected@edef\@do@newglossaryentry{%
4438         \noexpand\newglossaryentry{\the\glslabeltok}%
4439         {%
4440             type=\acronymtype,%
4441             short={\the\glssshorttok},%
4442             shortplural={\the\glssshorttok\noexpand\acrpluralsuffix},%
4443             long={\the\glslongtok},%
4444             longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4445             user1={\the\glssshorttok},%
4446             user2={\the\glssshorttok\noexpand\acrpluralsuffix},%

```

```

4447     user3={\the\glslongtok},%
4448     user4={\the\glslongtok\noexpand\acrpluralsuffix},%
4449     \CustomAcronymFields,%
4450     \the\glskeylisttok
4451   }%
4452 }%
4453 \@do@newglossaryentry
4454 }

```

## \SetCustomStyle

```

4455 \newcommand*{\SetCustomStyle}{%
4456   \renewcommand{\newacronym}[4][\{%
4457     \ifx\@glsacronymlists\@empty
4458       \def\@glo@type{\acronymtype}%
4459       \setkeys{glossentry}{##1}%
4460       \DeclareAcronymList{\@glo@type}%
4461       \SetCustomDisplayStyle{\@glo@type}%
4462     \fi
4463     \glskeylisttok{##1}%
4464     \glslabeltok{##2}%
4465     \glsshorttok{##3}%
4466     \glslongtok{##4}%
4467     \newacronymhook
4468     \CustomNewAcronymDef
4469   }%

```

Set the display

```

4470   \@for\@gls@type:=\@glsacronymlists\do{%
4471     \SetCustomDisplayStyle{\@gls@type}%
4472   }%
4473 }

```

## fineAcronymSynonyms

```

4474 \newcommand*{\DefineAcronymSynonyms}{%

```

### Short form

\acs

```

4475   \let\acs\acrshort

```

### First letter uppercase short form

\Acs

```

4476   \let\Acs\Acrshort

```

### Plural short form

\acsp

```

4477   \let\acsp\acrshortpl

```

### First letter uppercase plural short form

`\Acsp`  
 4478 `\let\Acsp\Acrshortpl`  
 Long form

`\acl`  
 4479 `\let\acl\acrlong`  
 Plural long form

`\aclp`  
 4480 `\let\aclp\acrlongpl`  
 First letter upper case long form

`\Acl`  
 4481 `\let\Acl\Acrlong`  
 First letter upper case plural long form

`\Aclp`  
 4482 `\let\Aclp\Acrlongpl`  
 Full form

`\acf`  
 4483 `\let\acf\acrfull`  
 Plural full form

`\acfp`  
 4484 `\let\acfp\acrfullpl`  
 First letter upper case full form

`\Acf`  
 4485 `\let\Acf\Acrfull`  
 First letter upper case plural full form

`\Acfp`  
 4486 `\let\Acfp\Acrfullpl`  
 Standard form

`\ac`  
 4487 `\let\ac\gls`  
 First upper case standard form

`\Ac`  
 4488 `\let\Ac\Gls`

Standard plural form

`\acp`

```
4489 \let\acp\glsp1
```

Standard first letter upper case plural form

`\Acp`

```
4490 \let\Acp\Glsp1
```

```
4491 }
```

Define synonyms if required

```
4492 \ifglacrshortcuts
```

```
4493 \DefineAcronymSynonyms
```

```
4494 \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.

```
4495 \RequirePackage{glossary-hypernav}
```

The styles that use list-like environments. These are not loaded if the `nolist` option is used:

```
4496 \@gls@loadlist
```

The styles that use the `longtable` environment. These are not loaded if the `no-long package` option is used.

```
4497 \@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.

```
4498 \@gls@loadsuper
```

The tree-like styles. These are not loaded if the `notree` package option is used.

```
4499 \@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.

```
4500 \ifx\@glossary@default@style\relax
```

```
4501 \else
```

```
4502 \glossarystyle{\@glossary@default@style}
```

```
4503 \fi
```

## 1.19 Debugging Commands

`\showgloparent`    `\showgloparent{<label>}`

```
4504 \newcommand*{\showgloparent}[1]{%  
4505   \expandafter\show\csname glo@#1@parent\endcsname  
4506 }
```

`\showglolevel`    `\showglolevel{<label>}`

```
4507 \newcommand*{\showglolevel}[1]{%  
4508   \expandafter\show\csname glo@#1@level\endcsname  
4509 }
```

`\showglotext`    `\showglotext{<label>}`

```
4510 \newcommand*{\showglotext}[1]{%  
4511   \expandafter\show\csname glo@#1@text\endcsname  
4512 }
```

`\showgloplural`    `\showgloplural{<label>}`

```
4513 \newcommand*{\showgloplural}[1]{%  
4514   \expandafter\show\csname glo@#1@plural\endcsname  
4515 }
```

`\showglofirst`    `\showglofirst{<label>}`

```
4516 \newcommand*{\showglofirst}[1]{%  
4517   \expandafter\show\csname glo@#1@first\endcsname  
4518 }
```

`\showglofirstpl`    `\showglofirstpl{<label>}`

```
4519 \newcommand*{\showglofirstpl}[1]{%  
4520   \expandafter\show\csname glo@#1@firstpl\endcsname  
4521 }
```

`\showglotype`    `\showglotype{<label>}`

```

4522 \newcommand*{\showglotype}[1]{%
4523   \expandafter\show\csname glo@#1@type\endcsname
4524 }

```

\showglocounter    \showglocounter{<label>}

```

4525 \newcommand*{\showglocounter}[1]{%
4526   \expandafter\show\csname glo@#1@counter\endcsname
4527 }

```

\showglouser    \showglouser{<label>}

```

4528 \newcommand*{\showglouser}[1]{%
4529   \expandafter\show\csname glo@#1@useri\endcsname
4530 }

```

\showglouserii    \showglouserii{<label>}

```

4531 \newcommand*{\showglouserii}[1]{%
4532   \expandafter\show\csname glo@#1@userii\endcsname
4533 }

```

\showglouseriii    \showglouseriii{<label>}

```

4534 \newcommand*{\showglouseriii}[1]{%
4535   \expandafter\show\csname glo@#1@useriii\endcsname
4536 }

```

\showglouseriv    \showglouseriv{<label>}

```

4537 \newcommand*{\showglouseriv}[1]{%
4538   \expandafter\show\csname glo@#1@useriv\endcsname
4539 }

```

\showglouserv    \showglouserv{<label>}

```

4540 \newcommand*{\showglouserv}[1]{%
4541   \expandafter\show\csname glo@#1@userv\endcsname
4542 }

```

\showglouservi    \showglouservi{<label>}

```
4543 \newcommand*{\showglouservi}[1]{%
4544   \expandafter\show\csname glo@#1@uservi\endcsname
4545 }
```

\showgloname    \showgloname{<label>}

```
4546 \newcommand*{\showgloname}[1]{%
4547   \expandafter\show\csname glo@#1@name\endcsname
4548 }
```

\showglodesc    \showglodesc{<label>}

```
4549 \newcommand*{\showglodesc}[1]{%
4550   \expandafter\show\csname glo@#1@desc\endcsname
4551 }
```

\showglodescplural    \showglodescplural{<label>}

```
4552 \newcommand*{\showglodescplural}[1]{%
4553   \expandafter\show\csname glo@#1@descplural\endcsname
4554 }
```

\showglosort    \showglosort{<label>}

```
4555 \newcommand*{\showglosort}[1]{%
4556   \expandafter\show\csname glo@#1@sort\endcsname
4557 }
```

\showglosymbol    \showglosymbol{<label>}

```
4558 \newcommand*{\showglosymbol}[1]{%
4559   \expandafter\show\csname glo@#1@symbol\endcsname
4560 }
```

\showglosymbolplural    \showglosymbolplural{<label>}

```
4561 \newcommand*{\showglosymbolplural}[1]{%
4562   \expandafter\show\csname glo@#1@symbolplural\endcsname
4563 }
```

`\showgloindex`    `\showgloindex{<label>}`

```
4564 \newcommand*{\showgloindex}[1]{%
4565   \expandafter\show\csname glo@#1@index\endcsname
4566 }
```

`\showgloflag`    `\showgloflag{<label>}`

```
4567 \newcommand*{\showgloflag}[1]{%
4568   \expandafter\show\csname ifglo@#1@flag\endcsname
4569 }
```

`\showacronymlists`    `\showacronymlists`

Show list of glossaries that have been flagged as a list of acronyms.

```
4570 \newcommand*{\showacronymlists}{%
4571   \show\@glsacronymlists
4572 }
```

`\showglossaries`    `\showglossaries`

Show list of defined glossaries.

```
4573 \newcommand*{\showglossaries}{%
4574   \show\@glo@types
4575 }
```

`\showglossaryin`    `\showglossaryin{<glossary-label>}`

Show the ‘in’ extension for the given glossary.

```
4576 \newcommand*{\showglossaryin}[1]{%
4577   \expandafter\show\csname @glotype@#1@in\endcsname
4578 }
```

`\showglossaryout`    `\showglossaryout{<glossary-label>}`

Show the ‘out’ extension for the given glossary.

```
4579 \newcommand*{\showglossaryout}[1]{%
4580   \expandafter\show\csname @glotype@#1@out\endcsname
4581 }
```

`\showglossarytitle`    `\showglossarytitle{<glossary-label>}`

Show the title for the given glossary.

```
4582 \newcommand*{\showglossarytitle}[1]{%
4583   \expandafter\show\csname @glotype@#1@title\endcsname
4584 }
```

`\showglossarycounter`    `\showglossarycounter{<glossary-label>}`

Show the counter for the given glossary.

```
4585 \newcommand*{\showglossarycounter}[1]{%
4586   \expandafter\show\csname @glotype@#1@counter\endcsname
4587 }
```

`\showglossaryentries`    `\showglossaryentries{<glossary-label>}`

Show the list of entry labels for the given glossary.

```
4588 \newcommand*{\showglossaryentries}[1]{%
4589   \expandafter\show\csname glolist@#1\endcsname
4590 }
```

## 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.

```
4591 \csname ifglcompatible-2.07\endcsname
4592   \RequirePackage{glossaries-compatible-207}
4593 \fi
```

## 2 Mfirstuc Documented Code

```
4594 \NeedsTeXFormat{LaTeX2e}
```

4595 \ProvidesPackage{mfirstuc}[2012/05/21 v1.06 (NLCT)]

Requires etoolbox:

4596 \RequirePackage{etoolbox}

\makefirstuc Syntax:

\makefirstuc{<text>}

Makes the first letter uppercase, but will skip initial control sequences if they are followed by a group and make the first thing in the group uppercase, unless the group is empty. Thus \makefirstuc{abc} will produce: Abc, \makefirstuc{\ae bc} will produce: Æbc, but \makefirstuc{\emph{abc}} will produce *Abc*. This is required by \Gls and \Glspl.

4597 \newif\if@glscs

4598 \newtoks\@glsmfirst

4599 \newtoks\@glsmrest

4600 \def\makefirstuc#1{%

4601 \def\gls@argi{#1}%

4602 \ifx\gls@argi\@empty

If the argument is empty, do nothing.

4603 \else

4604 \def\@gls@tmp{\ #1}%

4605 \@onelevel@sanitize\@gls@tmp

4606 \expandafter\@gls@checkcs\@gls@tmp\relax\relax

4607 \if@glscs

4608 \@gls@getbody #1{}\@nil

4609 \ifx\@gls@rest\@empty

4610 \glsmakefirstuc{#1}%

4611 \else

4612 \expandafter\@gls@split\@gls@rest\@nil

4613 \ifx\@gls@first\@empty

4614 \glsmakefirstuc{#1}%

4615 \else

4616 \expandafter\@glsmfirst\expandafter{\@gls@first}%

4617 \expandafter\@glsmrest\expandafter{\@gls@rest}%

4618 \edef\@gls@domfirstuc{\noexpand\@gls@body

4619 {\noexpand\glsmakefirstuc\the\@glsmfirst}%

4620 \the\@glsmrest}%

4621 \@gls@domfirstuc

4622 \fi

4623 \fi

4624 \else

4625 \glsmakefirstuc{#1}%

4626 \fi

4627 \fi

4628 }

Put first argument in \@gls@first and second argument in \@gls@rest:

4629 \def\@gls@split#1#2\@nil{%

```

4630 \def\@gls@first{#1}\def\@gls@rest{#2}%
4631 }

4632 \def\@gls@checkcs#1 #2#3\relax{%
4633 \def\@gls@argi{#1}\def\@gls@argii{#2}%
4634 \ifx\@gls@argi\@gls@argii
4635 \@gls@true
4636 \else
4637 \@gls@false
4638 \fi
4639 }

```

Make first thing upper case:

```

4640 \def\@gls@makefirstuc#1{\MakeUppercase #1}

```

`\glsmakefirstuc` Provide a user command to make it easier to customise.

```

4641 \newcommand*\glsmakefirstuc[1]{\@gls@makefirstuc{#1}}

```

Get the first grouped argument and stores in `\@gls@body`.

```

4642 \def\@gls@getbody#1#{\def\@gls@body{#1}\@gls@gobbletonil}

```

Scoop up everything to `\@nil` and store in `\@gls@rest`:

```

4643 \def\@gls@gobbletonil#1\@nil{\def\@gls@rest{#1}}

```

`\xmakefirstuc` Expand argument once before applying `\makefirstuc` (added v1.01).

```

4644 \newcommand*\xmakefirstuc[1]{%
4645 \expandafter\makefirstuc\expandafter{#1}}

```

`\capitalisewords` Capitalise each word in the argument. Words are considered to be separated by plain spaces (i.e. non-breakable spaces won't be considered a word break).

```

4646 \newcommand*\capitalisewords[1]{%
4647 \def\gls@add@space{}%
4648 \mfu@capitalisewords#1 \@nil\mfu@endcap
4649 %\gls@add@space\makefirstuc{##1}\def\gls@add@space{ }%
4650 }

```

```

4651 \def\mfu@capitalisewords#1 #2\mfu@endcap{%
4652 \def\mfu@cap@first{#1}%
4653 \def\mfu@cap@second{#2}%
4654 \gls@add@space
4655 \makefirstuc{#1}%
4656 \def\gls@add@space{ }%
4657 \ifx\mfu@cap@second\@nnil
4658 \let\next@mfu@cap\mfu@noop
4659 \else
4660 \let\next@mfu@cap\mfu@capitalisewords
4661 \fi
4662 \next@mfu@cap#2\mfu@endcap
4663 }
4664 \def\mfu@noop#1\mfu@endcap{}

```

`\xcapitalisewords` Short-cut command:

```
4665 \newcommand*\xcapitalisewords}[1]{%
4666   \expandafter\capitalisewords\expandafter{#1}%
4667 }
```

## 3 Glossary Styles

### 3.1 Glossary hyper-navigation definitions (glossary-hypernav package)

Package Definition:

```
4668 \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`

```
4669 \newcommand*\glsnavhyperlink}[3][\@glo@type]{%
4670   \edef\gls@grplabel{#2}\protected@edef\gls@grptitle{#3}%
4671   \@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`

```
4672 \newcommand*\glsnavhypertarget}[3][\@glo@type]{%
  Add this group to the aux file for re-run check.
4673   \protected@write\auxout{}\string\gls@hypergroup{#1}{#2}}%
  Add the target.
4674   \@glstarget{glsn:#1@#2}{#3}%
  Check list of know groups to determine if a re-run is required.
4675   \expandafter\let
4676     \expandafter\@gls@list\csname @gls@hypergroup@#1\endcsname
  Iterate through list and terminate loop if this group is found.
4677   \@for\@gls@elem:=\@gls@list\do{%
4678     \ifthenelse{\equal{\@gls@elem}{#2}}{\@endfortrue}{}}%
```

Check if list terminated prematurely.

```
4679 \if@endfor
4680 \else
```

This group was not included in the list, so issue a warning.

```
4681 \GlossariesWarningNoLine{Navigation panel
4682   for glossary type ‘#1’^^Jmissing group ‘#2’}%
4683 \gdef\gls@hypergrouprerun{%
4684   \GlossariesWarningNoLine{Navigation panel
4685   has changed. Rerun LaTeX}}%
4686 \fi
4687 }
```

`\gls@hypergrouprerun` Give a warning at the end if re-run required

```
4688 \let\gls@hypergrouprerun\relax
4689 \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.

```
4690 \newcommand*{\@gls@hypergroup}[2]{%
4691 \ifundefined{\@gls@hypergrouplist@#1}{%
4692   \expandafter\xdef\csname @gls@hypergrouplist@#1\endcsname{#2}%
4693 }{%
4694   \expandafter\let\expandafter\@gls@tmp
4695   \csname @gls@hypergrouplist@#1\endcsname
4696   \expandafter\xdef\csname @gls@hypergrouplist@#1\endcsname{%
4697     \@gls@tmp,#2}%
4698 }%
4699 }
```

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`

```
4700 \newcommand*{\glsnavigation}{%
4701 \def\@gls@between{%
4702 \@ifundefined{\@gls@hypergrouplist@\@glo@type}{%
4703   \def\@gls@list{%
4704 }{%
4705   \expandafter\let\expandafter\@gls@list
4706   \csname @gls@hypergrouplist@\@glo@type\endcsname
4707 }%}
```

```

4708 \@for\@gls@tmp:=\@gls@list\do{%
4709   \@gls@between
4710   \glsnavhyperlink{\@gls@tmp}{\glsgetgrouptitle{\@gls@tmp}}%
4711   \let\@gls@between\glshypernavsep%
4712 }%
4713 }

```

`\glshypernavsep` Separator for the hyper navigation bar.

```

4714 \newcommand*\glshypernavsep{\space\textbar\space}

```

The `\glssymbolnav` produces a simple navigation set of links for just the symbol and number groups. This used to be used at the start of `\glsnavigation`. This command is no longer needed.

`\glssymbolnav`

```

4715 \newcommand*\glssymbolnav{%
4716 \glsnavhyperlink{glssymbols}{\glsgetgrouptitle{glssymbols}}%
4717 \glshypernavsep
4718 \glsnavhyperlink{glsnumbers}{\glsgetgrouptitle{glsnumbers}}%
4719 \glshypernavsep
4720 }

```

### 3.2 In-line Style (glossary-inline.sty)

This defines an in-line style where the entries are comma-separated with just the name and description displayed.

```

4721 \ProvidesPackage{glossary-inline}[2012/09/21 v3.03 (NLCT)]

```

`inline` Define the inline style.

```

4722 \newglossarystyle{inline}{%

```

Start of glossary sets up first empty separator between entries. (This is then changed by `\glossaryentryfield`)

```

4723   \renewenvironment{theglossary}%
4724   {%
4725     \def\gls@inlinesep{}%
4726     \def\gls@inlinesubsep{}%
4727     \def\gls@inlinepostchild{}%
4728   }%
4729   {\glspostinline}%

```

No header:

```

4730 \renewcommand*\glossaryheader{}%

```

No group headings (if heading is required, add `\glsinlinedopostchild` to start definition in case heading follows a child entry):

```

4731 \renewcommand*\glsgroupheading[1]{}%

```

Just display separator followed by name and description:

```

4732 \renewcommand{\glossaryentryfield}[5]{%
4733   \glsinlinedopostchild
4734   \gls@inlinesep
4735   \def\glo@desc{##3}%
4736   \def\@no@post@desc{\nopostdesc}%
4737   \glstentryitem{##1}\glsinlinenameformat{##1}{##2}%
4738   \ifx\glo@desc\@no@post@desc
4739     \glsinlineemptydescformat{##4}{##5}%
4740   \else
4741     \ifstrempy{##3}%
4742       {\glsinlineemptydescformat{##4}{##5}}%
4743       {\glsinlinedescformat{##3}{##4}{##5}}%
4744   \fi
4745   \ifglshaschildren{##1}%
4746   {%
4747     \glsresetsubentrycounter
4748     \glsinlineparentchildseparator
4749     \def\gls@inlinesubsep{}%
4750     \def\gls@inlinepostchild{\glsinlinepostchild}%
4751   }%
4752   {%
4753   \def\gls@inlinesep{\glsinlineseparator}%
4754   }%

```

Sub-entries display description:

```

4755 \renewcommand{\glossarysubentryfield}[6]{%
4756   \gls@inlinesubsep%
4757   \glsinlinesubnameformat{##2}{##3}%
4758   \glssubentryitem{##2}\glsinlinesubdescformat{##4}{##5}{##6}%
4759   \def\gls@inlinesubsep{\glsinlinesubseparator}%
4760   }%

```

Nothing special between groups:

```

4761 \renewcommand*{\glsgroupskip}{;}%
4762 }

```

`\glsinlinedopostchild`

```

4763 \newcommand*{\glsinlinedopostchild}{%
4764   \gls@inlinepostchild
4765   \def\gls@inlinepostchild{}%
4766 }

```

`\glsinlineseparator` Separator to use between entries.

```

4767 \newcommand*{\glsinlineseparator}{;\space}

```

`\glsinlinesubseparator` Separator to use between sub-entries.

```

4768 \newcommand*{\glsinlinesubseparator}{,\space}

```

parentchildseparator    Separator to use between parent and children.  
4769 \newcommand\*{\glsinlineparentchildseparator}{:\space}

\glsinlinepostchild    Hook to use between child and next entry  
4770 \newcommand\*{\glsinlinepostchild}{}

\glspostinline    Terminator for inline glossary.  
4771 \newcommand\*{\glspostinline}{\glspostdescription\space}

\glsinlinenameformat    Formats the name of the entry (first argument label, second argument name):  
4772 \newcommand\*{\glsinlinenameformat}[2]{\glstarget{#1}{#2}}

\glsinlinedescformat    Formats the entry's description, symbol and location list:  
4773 \newcommand\*{\glsinlinedescformat}[3]{\space#1}

\glsinlineemptydescformat    Formats the entry's symbol and location list when the description is empty:  
4774 \newcommand\*{\glsinlineemptydescformat}[2]{}

\glsinlinesubnameformat    Formats the name of the subentry (first argument label, second argument name):  
4775 \newcommand\*{\glsinlinesubnameformat}[2]{\glstarget{#1}{}}

\glsinlinesubdescformat    Formats the subentry's description, symbol and location list:  
4776 \newcommand\*{\glsinlinesubdescformat}[3]{#1}

### 3.3 List Style (glossary-list.sty)

The style file defines glossary styles that use the description environment. Note that since the entry name is placed in the optional argument to the `\item` command, it will appear in a bold font by default.

4777 \ProvidesPackage{glossary-list}[2012/11/11 v3.04 (NLCT)]

**list**    The list glossary style uses the description environment. The group separator `\glsgroupskip` is redefined as `\indexspace` which produces a gap between groups. The glossary heading and the group headings do nothing. Sub-entries immediately follow the main entry without the sub-entry name. This style does not use the entry's symbol. This is used as the default style for the glossaries package.

4778 \newglossarystyle{list}{%

    Use description environment:

4779    \renewenvironment{theglossary}%

4780      {\begin{description}}{\end{description}}%

    No header at the start of the environment:

4781    \renewcommand\*{\glossaryheader}{}%

No group headings:

```
4782 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries start a new item in the list:

```
4783 \renewcommand*{\glossaryentryfield}[5]{%
4784 \item[\glsentryitem{##1}\glstarget{##1}{##2}]
4785 ##3\glspostdescription\space ##5}%
```

Sub-entries continue on the same line:

```
4786 \renewcommand*{\glossarysubentryfield}[6]{%
4787 \glssubentryitem{##2}%
4788 \glstarget{##2}{\strut}##4\glspostdescription\space ##6.}%
4789 % \end{macrocode}
4790 % Add vertical space between groups:
4791 %\changes{3.03}{2012/09/21}{added check for glsnogroupskip}
4792 % \begin{macrocode}
4793 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
4794 }
```

**listgroup** The listgroup style is like the list style, but the glossary groups have headings.

```
4795 \newglossarystyle{listgroup}{%
```

Base it on the list style:

```
4796 \glossarystyle{list}%
```

Each group has a heading:

```
4797 \renewcommand*{\glsgroupheading}[1]{\item[\glsgetgrouptitle{##1}]}}
```

**listhypergroup** The listhypergroup style is like the listgroup style, but has a set of links to the groups at the start of the glossary.

```
4798 \newglossarystyle{listhypergroup}{%
```

Base it on the list style:

```
4799 \glossarystyle{list}%
```

Add navigation links at the start of the environment:

```
4800 \renewcommand*{\glossaryheader}{%
4801 \item[\glsnavigation]}%
```

Each group has a heading with a hypertarget:

```
4802 \renewcommand*{\glsgroupheading}[1]{%
4803 \item[\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}]}}
```

**altlist** The altlist glossary style is like the list style, but places the description on a new line. Sub-entries follow in separate paragraphs without the sub-entry name. This style does not use the entry's symbol.

```
4804 \newglossarystyle{altlist}{%
```

Base it on the list style:

```
4805 \glossarystyle{list}%
```

Main (level 0) entries start a new item in the list with a line break after the entry name:

```
4806 \renewcommand*{\glossaryentryfield}[5]{%
4807   \item[\glstentryitem{##1}\glstarget{##1}{##2}]%
```

Version 3.04 changed `\newline` to the following paragraph break stuff (thanks to Daniel Gebhardt for supplying the fix) to prevent a page break occurring at this point.

```
4808   \mbox{ }\par\nobreak\@afterheading
4809   ##3\glspostdescription\space ##5)%
```

Sub-entries start a new paragraph:

```
4810 \renewcommand{\glossarysubentryfield}[6]{%
4811   \par
4812   \glssubentryitem{##2}%
4813   \glstarget{##2}{\strut}##4\glspostdescription\space ##6)%
4814 }
```

**altlistgroup** The altlistgroup glossary style is like the altlist style, but the glossary groups have headings.

```
4815 \newglossarystyle{altlistgroup}{%
```

Base it on the altlist style:

```
4816 \glossarystyle{altlist}%
```

Each group has a heading:

```
4817 \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.

```
4818 \newglossarystyle{altlisthypergroup}{%
```

Base it on the altlist style:

```
4819 \glossarystyle{altlist}%
```

Add navigation links at the start of the environment:

```
4820 \renewcommand*{\glossaryheader}{%
```

```
4821   \item[\glsnavigation]}%
```

Each group has a heading with a hypertarget:

```
4822 \renewcommand*{\glsgroupheading}[1]{%
```

```
4823   \item[\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}]}}
```

**listdotted** The listdotted glossary style was supplied by Axel Menzel. I've modified it slightly so that the distance from the start of the name to the end of the dotted line is specified by `\glslistdottedwidth`. Note that this style ignores the page numbers as well as the symbol. Sub-entries are displayed in the same way as top-level entries.

```
4824 \newglossarystyle{listdotted}{%
```

Base it on the list style:

```
4825 \glossarystyle{list}%
```

Each main (level 0) entry starts a new item:

```
4826 \renewcommand*{\glossaryentryfield}[5]{%
4827   \item[]\makebox[\glslistdottedwidth][l]{%
4828     \glssentryitem{##1}\glstarget{##1}{##2}%
4829     \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##3}%
```

Sub entries have the same format as main entries:

```
4830 \renewcommand*{\glossarysubentryfield}[6]{%
4831   \item[]\makebox[\glslistdottedwidth][l]{%
4832     \glssubentryitem{##2}%
4833     \glstarget{##2}{##3}%
4834     \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##4}%
4835 }
```

`\glslistdottedwidth`

```
4836 \newlength\glslistdottedwidth
4837 \setlength{\glslistdottedwidth}{.5\hsize}
```

`sublistdotted` This style is similar to the `glostylelistdotted` style, except that the main entries just have the name displayed.

```
4838 \newglossarystyle{sublistdotted}{%
```

Base it on the `listdotted` style:

```
4839 \glossarystyle{listdotted}%
```

Main (level 0) entries just display the name:

```
4840 \renewcommand*{\glossaryentryfield}[5]{%
4841   \item[\glssentryitem{##1}\glstarget{##1}{##2}]}%
4842 }
```

### 3.4 Glossary Styles using `longtable` (the `glossary-long` package)

The glossary styles defined in the package used the `longtable` environment in the glossary.

```
4843 \ProvidesPackage{glossary-long}[2012/09/21 v3.03 (NLCT)]
```

Requires the package:

```
4844 \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`.)

```
4845 \@ifundefined{glsdescwidth}{%
4846   \newlength\glsdescwidth
4847   \setlength{\glsdescwidth}{0.6\hsize}
4848 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column.

```
4849 \@ifundefined{glspagelistwidth}{%  
4850   \newlength{glspagelistwidth}  
4851   \setlength{glspagelistwidth}{0.1\hsize}  
4852 }{}
```

`long` The long glossary style command which uses the longtable environment:

```
4853 \newglossarystyle{long}{%
```

Use longtable with two columns:

```
4854   \renewenvironment{theglossary}{%  
4855     {\begin{longtable}{\lp{glstdescwidth}}}%  
4856     {\end{longtable}}%
```

Do nothing at the start of the environment:

```
4857   \renewcommand*{\glossaryheader}{}%
```

No heading between groups:

```
4858   \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries displayed in a row:

```
4859   \renewcommand*{\glossaryentryfield}[5]{%  
4860     \glstryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\}%
```

Sub entries displayed on the following row without the name:

```
4861   \renewcommand*{\glossarysubentryfield}[6]{%  
4862     &  
4863     \glssubentryitem{##2}%  
4864     \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\}%
```

Blank row between groups:

```
4865   \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & \\fi}%  
4866 }
```

`longborder` The longborder style is like the above, but with horizontal and vertical lines:

```
4867 \newglossarystyle{longborder}{%
```

Base it on the `glostylelong` style:

```
4868   \glossarystyle{long}%
```

Use longtable with two columns with vertical lines between each column:

```
4869   \renewenvironment{theglossary}{%  
4870     \begin{longtable}{\lllp{glstdescwidth}|}}{\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
4871   \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%  
4872 }
```

`longheader` The longheader style is like the long style but with a header:

```
4873 \newglossarystyle{longheader}{%
```

Base it on the `glostylelong` style:

```
4874   \glossarystyle{long}%
```

Set the table's header:

```
4875 \renewcommand*{\glossaryheader}{%
4876     \bfseries \entryname & \bfseries \descriptionname\\\endhead}%
4877 }
```

`longheaderborder` The `longheaderborder` style is like the `long` style but with a header and border:

```
4878 \newglossarystyle{longheaderborder}{%
```

Base it on the `glostylelongborder` style:

```
4879 \glossarystyle{longborder}%
```

Set the table's header and add horizontal line to table's foot:

```
4880 \renewcommand*{\glossaryheader}{%
4881     \hline\bfseries \entryname & \bfseries \descriptionname\\\hline
4882     \endhead
4883     \hline\endfoot}%
4884 }
```

`long3col` The `long3col` style is like `long` but with 3 columns

```
4885 \newglossarystyle{long3col}{%
```

Use a `longtable` with 3 columns:

```
4886 \renewenvironment{theglossary}%
4887     {\begin{longtable}{lp{\glstdescwidth}p{\glspagelistwidth}}}%
4888     {\end{longtable}}%
```

No table header:

```
4889 \renewcommand*{\glossaryheader}{}%
```

No headings between groups:

```
4890 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
4891 \renewcommand*{\glossaryentryfield}[5]{%
4892     \glstarget{##1}{\glstarget{##1}{##2} & ##3 & ##5\\}%
```

Sub-entries on a separate row (no name, description in second column, page list in third column):

```
4893 \renewcommand*{\glossarysubentryfield}[6]{%
4894     &
4895     \glssubentryitem{##2}%
4896     \glstarget{##2}{\strut}##4 & ##6\\}%
```

Blank row between groups:

```
4897 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & &\\\fi}%
4898 }
```

`long3colborder` The `long3colborder` style is like the `long3col` style but with a border:

```
4899 \newglossarystyle{long3colborder}{%
```

Base it on the `glostylelong3col` style:

```
4900 \glossarystyle{long3col}%
```

Use a `longtable` with 3 columns with vertical lines around them:

```
4901 \renewenvironment{theglossary}%  
4902 {\begin{longtable}{|l|p{\glstdescwidth}|p{\glspagelistwidth|}}%  
4903 {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
4904 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%  
4905 }
```

`long3colheader` The `long3colheader` style is like `long3col` but with a header row:

```
4906 \newglossarystyle{long3colheader}{%
```

Base it on the `glostylelong3col` style:

```
4907 \glossarystyle{long3col}%
```

Set the table's header:

```
4908 \renewcommand*{\glossaryheader}{%  
4909 \bfseries\entryname&\bfseries\descriptionname&  
4910 \bfseries\pagelistname\\\endhead}%  
4911 }
```

`long3colheaderborder` The `long3colheaderborder` style is like the above but with a border

```
4912 \newglossarystyle{long3colheaderborder}{%
```

Base it on the `glostylelong3colborder` style:

```
4913 \glossarystyle{long3colborder}%
```

Set the table's header and add horizontal line at table's foot:

```
4914 \renewcommand*{\glossaryheader}{%  
4915 \hline  
4916 \bfseries\entryname&\bfseries\descriptionname&  
4917 \bfseries\pagelistname\\\hline\endhead  
4918 \hline\endfoot}%  
4919 }
```

`long4col` The `long4col` style has four columns where the third column contains the value of the associated symbol key.

```
4920 \newglossarystyle{long4col}{%
```

Use a `longtable` with 4 columns:

```
4921 \renewenvironment{theglossary}%  
4922 {\begin{longtable}{|llll|}}%  
4923 {\end{longtable}}%
```

No table header:

```
4924 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
4925 \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):

```
4926 \renewcommand*{\glossaryentryfield}[5]{%
4927   \glstentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\\}%
```

Sub entries on a single row with no name (description in second column, symbol in third column, page list in last column):

```
4928 \renewcommand*{\glossarysubentryfield}[6]{%
4929   &
4930   \glssubentryitem{##2}%
4931   \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%
```

Blank row between groups:

```
4932 \renewcommand*{\glsnoglobskip}{\ifglsnogroupskip\else & & \\fi}%
4933 }
```

**long4colheader** The long4colheader style is like long4col but with a header row.

```
4934 \newglossarystyle{long4colheader}{%
```

Base it on the glostylelong4col style:

```
4935 \glossarystyle{long4col}%
```

Table has a header:

```
4936 \renewcommand*{\glossaryheader}{%
4937   \bfseries\entryname&\bfseries\descriptionname&
4938   \bfseries \symbolname&
4939   \bfseries\pagelistname\\\endhead}%
4940 }
```

**long4colborder** The long4colborder style is like long4col but with a border.

```
4941 \newglossarystyle{long4colborder}{%
```

Base it on the glostylelong4col style:

```
4942 \glossarystyle{long4col}%
```

Use a longtable with 4 columns surrounded by vertical lines:

```
4943 \renewenvironment{theglossary}%
4944   {\begin{longtable}{|l|l|l|l|}}%
4945   {\end{longtable}}%
```

Add horizontal lines to the head and foot of the table:

```
4946 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
4947 }
```

**long4colheaderborder** The long4colheaderborder style is like the above but with a border.

```
4948 \newglossarystyle{long4colheaderborder}{%
```

Base it on the glostylelong4col style:

```
4949 \glossarystyle{long4col}%
```

Use a longtable with 4 columns surrounded by vertical lines:

```
4950 \renewenvironment{theglossary}%
4951   {\begin{longtable}{|l|l|l|l|}}%
4952   {\end{longtable}}%
```

Add table header and horizontal line at the table's foot:

```
4953 \renewcommand*{\glossaryheader}{%
4954   \hline\bfseries\entryname&\bfseries\descriptionname&
4955   \bfseries \symbolname&
4956   \bfseries\pagelistname\\ \hline\endhead\hline\endfoot}%
4957 }
```

**altlong4col** The altlong4col style is like the long4col style but can have multiline descriptions and page lists.

```
4958 \newglossarystyle{altlong4col}{%
```

Base it on the glostylelong4col style:

```
4959   \glossarystyle{long4col}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
4960 \renewenvironment{theglossary}%
4961   {\begin{longtable}{lp{\glsdescwidth}lp{\glspagelistwidth}}}%
4962   {\end{longtable}}%
4963 }
```

**altlong4colheader** The altlong4colheader style is like altlong4col but with a header row.

```
4964 \newglossarystyle{altlong4colheader}{%
```

Base it on the glostylelong4colheader style:

```
4965   \glossarystyle{long4colheader}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
4966 \renewenvironment{theglossary}%
4967   {\begin{longtable}{lp{\glsdescwidth}lp{\glspagelistwidth}}}%
4968   {\end{longtable}}%
4969 }
```

**altlong4colborder** The altlong4colborder style is like altlong4col but with a border.

```
4970 \newglossarystyle{altlong4colborder}{%
```

Base it on the glostylelong4colborder style:

```
4971   \glossarystyle{long4colborder}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
4972 \renewenvironment{theglossary}%
4973   {\begin{longtable}{|l|lp{\glsdescwidth}|lp{\glspagelistwidth}|}%
4974   {\end{longtable}}%
4975 }
```

ong4colheaderborder The altlong4colheaderborder style is like the above but with a header as well as a border.

```
4976 \newglossarystyle{altlong4colheaderborder}{%
```

Base it on the glostylelong4colheaderborder style:

```
4977 \glossarystyle{long4colheaderborder}{%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
4978 \renewenvironment{theglossary}{%
```

```
4979 {\begin{longtable}{|l|p{\glsdescwidth}|l|p{\glspagelistwidth}|}}%
```

```
4980 {\end{longtable}}%
```

```
4981 }
```

### 3.5 Glossary Styles using longtable (the glossary-longragged package)

The glossary styles defined in the package used the longtable environment in the glossary and use ragged right formatting for the multiline columns.

```
4982 \ProvidesPackage{glossary-longragged}[2012/09/21 v3.03 (NLCT)]
```

Requires the package:

```
4983 \RequirePackage{array}
```

Requires the package:

```
4984 \RequirePackage{longtable}
```

\glsdescwidth This is a length that governs the width of the description column. This may have already been defined.

```
4985 \@ifundefined{glsdescwidth}{%
```

```
4986 \newlength\glsdescwidth
```

```
4987 \setlength{\glsdescwidth}{0.6\hsize}
```

```
4988 }{}
```

\glspagelistwidth This is a length that governs the width of the page list column. This may already have been defined.

```
4989 \@ifundefined{glspagelistwidth}{%
```

```
4990 \newlength\glspagelistwidth
```

```
4991 \setlength{\glspagelistwidth}{0.1\hsize}
```

```
4992 }{}
```

longragged The longragged glossary style is like the long but uses ragged right formatting for the description column.

```
4993 \newglossarystyle{longragged}{%
```

Use longtable with two columns:

```
4994 \renewenvironment{theglossary}{%
```

```
4995 {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}}}%
```

```
4996 {\end{longtable}}%
```

Do nothing at the start of the environment:

```
4997 \renewcommand*{\glossaryheader}{}%
```

No heading between groups:

```
4998 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries displayed in a row:

```
4999 \renewcommand*{\glossaryentryfield}[5]{%
5000     \glstentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%
5001     \tabularnewline}%
```

Sub entries displayed on the following row without the name:

```
5002 \renewcommand*{\glossarysubentryfield}[6]{%
5003     &
5004     \glssubentryitem{##2}%
5005     \glstarget{##2}{\strut}##4\glspostdescription\space ##6%
5006     \tabularnewline}%
```

Blank row between groups:

```
5007 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & \tabularnewline\fi}%
5008 }
```

`longraggedborder` The `longraggedborder` style is like the above, but with horizontal and vertical lines:

```
5009 \newglossarystyle{longraggedborder}{%
```

Base it on the `glostylelongragged` style:

```
5010 \glossarystyle{longragged}%
```

Use `longtable` with two columns with vertical lines between each column:

```
5011 \renewenvironment{theglossary}{%
5012     \begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|}%
5013     {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
5014 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
5015 }
```

`longraggedheader` The `longraggedheader` style is like the `longragged` style but with a header:

```
5016 \newglossarystyle{longraggedheader}{%
```

Base it on the `glostylelongragged` style:

```
5017 \glossarystyle{longragged}%
```

Set the table's header:

```
5018 \renewcommand*{\glossaryheader}{%
5019     \bfseries \entryname & \bfseries \descriptionname
5020     \tabularnewline\endhead}%
5021 }
```

`raggedheaderborder` The `longraggedheaderborder` style is like the `longragged` style but with a header and border:

```
5022 \newglossarystyle{longraggedheaderborder}{%
```

Base it on the `glostylelongraggedborder` style:

```
5023 \glossarystyle{longraggedborder}{%
```

Set the table's header and add horizontal line to table's foot:

```
5024 \renewcommand*{\glossaryheader}{%
5025 \hline\bfseries \entryname & \bfseries \descriptionname
5026 \tabularnewline\hline
5027 \endhead
5028 \hline\endfoot}%
5029 }
```

`longragged3col` The `longragged3col` style is like `longragged` but with 3 columns

```
5030 \newglossarystyle{longragged3col}{%
```

Use a longtable with 3 columns:

```
5031 \renewenvironment{theglossary}%
5032 {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}%
5033 >{\raggedright}p{\glspagelistwidth}}}%
5034 {\end{longtable}}%
```

No table header:

```
5035 \renewcommand*{\glossaryheader}{}%
```

No headings between groups:

```
5036 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
5037 \renewcommand*{\glossaryentryfield}[5]{%
5038 \glsentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\tabularnewline}%
```

Sub-entries on a separate row (no name, description in second column, page list in third column):

```
5039 \renewcommand*{\glossarysubentryfield}[6]{%
5040 &
5041 \glssubentryitem{##2}%
5042 \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%
```

Blank row between groups:

```
5043 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & \&\tabularnewline\fi}%
5044 }
```

`longragged3colborder` The `longragged3colborder` style is like the `longragged3col` style but with a border:

```
5045 \newglossarystyle{longragged3colborder}{%
```

Base it on the `glostylelongragged3col` style:

```
5046 \glossarystyle{longragged3col}%
```

Use a longtable with 3 columns with vertical lines around them:

```
5047 \renewenvironment{theglossary}%
5048 {\begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|}%
```

```

5049      >{\raggedright}p{\glspagelistwidth}{|}}%
5050    {\end{longtable}}}%

```

Place horizontal lines at the head and foot of the table:

```

5051    \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
5052 }

```

**longragged3colheader** The longragged3colheader style is like longragged3col but with a header row:

```

5053 \newglossarystyle{longragged3colheader}{%

```

Base it on the glostylelongragged3col style:

```

5054   \glossarystyle{longragged3col}%

```

Set the table's header:

```

5055   \renewcommand*{\glossaryheader}{%
5056     \bfseries\entryname&\bfseries\descriptionname&
5057     \bfseries\pagelistname\tabularnewline\endhead}%
5058 }

```

**longragged3colheaderborder** The longragged3colheaderborder style is like the above but with a border

```

5059 \newglossarystyle{longragged3colheaderborder}{%

```

Base it on the glostylelongragged3colborder style:

```

5060   \glossarystyle{longragged3colborder}%

```

Set the table's header and add horizontal line at table's foot:

```

5061   \renewcommand*{\glossaryheader}{%
5062     \hline
5063     \bfseries\entryname&\bfseries\descriptionname&
5064     \bfseries\pagelistname\tabularnewline\hline\endhead
5065     \hline\endfoot}%
5066 }

```

**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.

```

5067 \newglossarystyle{altlongragged4col}{%

```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```

5068   \renewenvironment{theglossary}%
5069     {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%
5070       >{\raggedright}p{\glspagelistwidth}}}%
5071     {\end{longtable}}}%

```

No table header:

```

5072   \renewcommand*{\glossaryheader}{}%

```

No group headings:

```

5073   \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):

```
5074 \renewcommand*{\glossaryentryfield}[5]{%
5075   \glstentryitem{##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):

```
5076 \renewcommand*{\glossarysubentryfield}[6]{%
5077   &
5078   \glssubentryitem{##2}%
5079   \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%
```

Blank row between groups:

```
5080 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & & \tabularnewline\fi}%
5081 }
```

`ongragged4colheader` The `altlongragged4colheader` style is like `altlongragged4col` but with a header row.

```
5082 \newglossarystyle{altlongragged4colheader}{%
```

Base it on the `glostylealtlongragged4col` style:

```
5083 \glossarystyle{altlongragged4col}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
5084 \renewenvironment{theglossary}%
5085   {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%
5086    >{\raggedright}p{\glspagelistwidth}}}%
5087   {\end{longtable}}%
```

Table has a header:

```
5088 \renewcommand*{\glossaryheader}{%
5089   \bfseries\entryname&\bfseries\descriptionname&
5090   \bfseries \symbolname&
5091   \bfseries\pagelistname\tabularnewline\endhead}%
5092 }
```

`ongragged4colborder` The `altlongragged4colborder` style is like `altlongragged4col` but with a border.

```
5093 \newglossarystyle{altlongragged4colborder}{%
```

Base it on the `glostylealtlongragged4col` style:

```
5094 \glossarystyle{altlongragged4col}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
5095 \renewenvironment{theglossary}%
5096   {\begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|l|%
5097    >{\raggedright}p{\glspagelistwidth}|}%
5098   {\end{longtable}}%
```

Add horizontal lines to the head and foot of the table:

```
5099 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
5100 }
```

`ged4colheaderborder` The `altlongragged4colheaderborder` style is like the above but with a header as well as a border.

```
5101 \newglossarystyle{altlongragged4colheaderborder}{%
```

Base it on the `glostylealtlongragged4col` style:

```
5102 \glossarystyle{altlongragged4col}%
```

Use a longtable with 4 columns where the second and last columns may have multiple lines in each row:

```
5103 \renewenvironment{theglossary}%
5104 {\begin{longtable}{|l|>{\raggedright}p{\glstdescwidth}|l|}%
5105 >{\raggedright}p{\glspagelistwidth}|}%
5106 {\end{longtable}}%
```

Add table header and horizontal line at the table's foot:

```
5107 \renewcommand*{\glossaryheader}{%
5108 \hline\bfseries\entryname&\bfseries\descriptionname&
5109 \bfseries \symbolname&
5110 \bfseries\pagelistname\tabularnewline\hline\endhead
5111 \hline\endfoot}%
5112 }
```

### 3.6 Glossary Styles using multicol (glossary-mcols.sty)

The style file defines glossary styles that use the `multicol` package. These use the tree-like glossary styles in a `multicol` environment.

```
5113 \ProvidesPackage{glossary-mcols}[2013/04/21 v3.05 (NLCT)]
```

Required packages:

```
5114 \RequirePackage{multicol}
5115 \RequirePackage{glossary-tree}
```

`\glsmcols` Define macro in which to store the number of columns. (Defaults to 2.)

```
5116 \newcommand*{\glsmcols}{2}
```

`mcolindex` Multi-column index style. Same as the index, but puts the glossary in multiple columns. (Ideally the glossary title should go in the optional argument of `multicols`, but the title isn't part of the glossary style.)

```
5117 \newglossarystyle{mcolindex}{%
5118 \glossarystyle{index}%
5119 \renewenvironment{theglossary}%
5120 {%
```

```

5121 \begin{multicols}{\glsmcols}
5122 \setlength{\parindent}{0pt}%
5123 \setlength{\parskip}{0pt plus 0.3pt}%
5124 \let\item\@idxitem}%
5125 {\end{multicols}}%
5126 }

```

**mcolindexgroup** As mcolindex but has headings:

```

5127 \newglossarystyle{mcolindexgroup}{%
5128 \glossarystyle{mcolindex}%
5129 \renewcommand*{\glsgroupheading}[1]{%
5130 \item\textbf{\glsgroupheading{##1}}\indexspace}%
5131 }

```

**mcolindexhypergroup** The mcolindexhypergroup style is like the mcolindexgroup style but has hyper navigation.

```

5132 \newglossarystyle{mcolindexhypergroup}{%
    Base it on the glostylemcolindex style:
5133 \glossarystyle{mcolindex}%
    Put navigation links to the groups at the start of the glossary:
5134 \renewcommand*{\glossaryheader}{%
5135 \item\textbf{\glshypernavigation}\indexspace}%
    Add a heading for each group (with a target). The group's title is in bold followed
    by a vertical gap.
5136 \renewcommand*{\glsgroupheading}[1]{%
5137 \item\textbf{\glshypernavhypertarget{##1}}{\glsgroupheading{##1}}}%
5138 \indexspace}%
5139 }

```

**mcoltree** Multi-column index style. Same as the tree, but puts the glossary in multiple columns.

```

5140 \newglossarystyle{mcoltree}{%
5141 \glossarystyle{tree}%
5142 \renewenvironment{theglossary}%
5143 {%
5144 \begin{multicols}{\glsmcols}
5145 \setlength{\parindent}{0pt}%
5146 \setlength{\parskip}{0pt plus 0.3pt}%
5147 }%
5148 {\end{multicols}}%
5149 }

```

**mcoltreegroup** Like the mcoltree style but the glossary groups have headings.

```

5150 \newglossarystyle{mcoltreegroup}{%
    Base it on the glostylemcoltree style:
5151 \glossarystyle{mcoltree}%

```

Each group has a heading (in bold) followed by a vertical gap):

```
5152 \renewcommand{\glsgroupheading}[1]{\par
5153 \noindent\textbf{\glsgetgrouptitle{##1}}\par\indexspace}%
5154 }
```

**mcoltreehypergroup** The mcoltreehypergroup style is like the treegroup style, but has a set of links to the groups at the start of the glossary.

```
5155 \newglossarystyle{mcoltreehypergroup}{%
```

Base it on the glostylemcoltree style:

```
5156 \glossarystyle{mcoltree}%
```

Put navigation links to the groups at the start of the theglossary environment:

```
5157 \renewcommand*{\glossaryheader}{%
5158 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
5159 \renewcommand*{\glsgroupheading}[1]{%
5160 \par\noindent
5161 \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
5162 \indexspace}%
5163 }
```

**mcoltreenoname** Multi-column index style. Same as the treenoname, but puts the glossary in multiple columns.

```
5164 \newglossarystyle{mcoltreenoname}{%
5165 \glossarystyle{treenoname}%
5166 \renewenvironment{theglossary}%
5167 {%
5168 \begin{multicols}{\glsmcols}
5169 \setlength{\parindent}{0pt}%
5170 \setlength{\parskip}{0pt plus 0.3pt}%
5171 }%
5172 {\end{multicols}}}%
5173 }
```

**mcoltreenonamegroup** Like the mcoltreenoname style but the glossary groups have headings.

```
5174 \newglossarystyle{mcoltreenonamegroup}{%
```

Base it on the glostylemcoltreenoname style:

```
5175 \glossarystyle{mcoltreenoname}%
```

Give each group a heading:

```
5176 \renewcommand{\glsgroupheading}[1]{\par
5177 \noindent\textbf{\glsgetgrouptitle{##1}}\par\indexspace}%
5178 }
```

**treenonamehypergroup** The mcoltreenonamehypergroup style is like the mcoltreenonamegroup style, but has a set of links to the groups at the start of the glossary.

```
5179 \newglossarystyle{mcoltreenonamehypergroup}{%
```

Base it on the `glostylemcoltreename` style:

```
5180 \glossarystyle{mcoltreename}%
```

Put navigation links to the groups at the start of the `theglossary` environment:

```
5181 \renewcommand*{\glossaryheader}{%
```

```
5182 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
5183 \renewcommand*{\glsgroupheading}[1]{%
```

```
5184 \par\noindent
```

```
5185 \textbf{\glsnavigationhypertarget{##1}{\glsgroupheading{##1}}}\par
```

```
5186 \indexspace}%
```

```
5187 }
```

`mcolalttree` Multi-column index style. Same as the `alttree`, but puts the glossary in multiple columns.

```
5188 \newglossarystyle{mcolalttree}{%
```

```
5189 \glossarystyle{alttree}%
```

```
5190 \renewenvironment{theglossary}%
```

```
5191 {%
```

```
5192 \begin{multicols}{\glsmcols}
```

```
5193 \def\@gls@prevlevel{-1}%
```

```
5194 \mbox{}\par
```

```
5195 }%
```

```
5196 {\par\end{multicols}}%
```

```
5197 }
```

`mcolalttreegroup` Like the `mcolalttree` style but the glossary groups have headings.

```
5198 \newglossarystyle{mcolalttreegroup}{%
```

Base it on the `glostylemcolalttree` style:

```
5199 \glossarystyle{mcolalttree}%
```

Give each group a heading.

```
5200 \renewcommand{\glsgroupheading}[1]{\par
```

```
5201 \def\@gls@prevlevel{-1}%
```

```
5202 \hangindent0pt\relax
```

```
5203 \parindent0pt\relax
```

```
5204 \textbf{\glsgroupheading{##1}}\par\indexspace}%
```

```
5205 }
```

`olalttreehypergroup` The `mcolalttreehypergroup` style is like the `mcolalttreegroup` style, but has a set of links to the groups at the start of the glossary.

```
5206 \newglossarystyle{mcolalttreehypergroup}{%
```

Base it on the `glostylemcolalttree` style:

```
5207 \glossarystyle{mcolalttree}%
```

Put the navigation links in the header

```
5208 \renewcommand*{\glossaryheader}{%
5209   \par
5210   \def\@gls@prevlevel{-1}%
5211   \hangindent0pt\relax
5212   \parindent0pt\relax
5213   \textbf{\glsnavigation}\par\indexspace}%
```

Put a hypertarget at the start of each group

```
5214 \renewcommand*{\glsgroupheading}[1]{%
5215   \par
5216   \def\@gls@prevlevel{-1}%
5217   \hangindent0pt\relax
5218   \parindent0pt\relax
5219   \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
5220   \indexspace}}
```

### 3.7 Glossary Styles using supertabular environment (glossary-super package)

The glossary styles defined in the package use the supertabular environment.

```
5221 \ProvidesPackage{glossary-super}[2012/09/21 v3.03 (NLCT)]
```

Requires the package:

```
5222 \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.

```
5223 \@ifundefined{glsdescwidth}{%
5224   \newlength\glsdescwidth
5225   \setlength{\glsdescwidth}{0.6\hsize}
5226 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined if has been loaded.

```
5227 \@ifundefined{glspagelistwidth}{%
5228   \newlength\glspagelistwidth
5229   \setlength{\glspagelistwidth}{0.1\hsize}
5230 }{}
```

`super` The super glossary style uses the supertabular environment (it uses lengths defined in the package.)

```
5231 \newglossarystyle{super}{%
```

Put the glossary in a supertabular environment with two columns and no head or tail:

```
5232   \renewenvironment{theglossary}{%
5233     {\tablehead{}\tabletail{}}%
5234     \begin{supertabular}{lp{\glsdescwidth}}}%
5235     {\end{supertabular}}%
```

Do nothing at the start of the table:

```
5236 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
5237 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries put in a row (name in first column, description and page list in second column):

```
5238 \renewcommand*{\glossaryentryfield}[5]{}%
```

```
5239 \glstryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\}%
```

Sub entries put in a row (no name, description and page list in second column):

```
5240 \renewcommand*{\glossarysubentryfield}[6]{}%
```

```
5241 &
```

```
5242 \glssubentryitem{##2}%
```

```
5243 \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\}%
```

Blank row between groups:

```
5244 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & \\fi}%
```

```
5245 }
```

**superborder** The superborder style is like the above, but with horizontal and vertical lines:

```
5246 \newglossarystyle{superborder}{}%
```

Base it on the `glostylessuper` style:

```
5247 \glossarystyle{super}%
```

Put the glossary in a `supertabular` environment with two columns and a horizontal line in the head and tail:

```
5248 \renewenvironment{theglossary}%
```

```
5249 {\tablehead{\hline}\tabletail{\hline}%
```

```
5250 \begin{supertabular}{|l|p{\glstdescwidth}|}%
```

```
5251 {\end{supertabular}}%
```

```
5252 }
```

**superheader** The superheader style is like the `super` style, but with a header:

```
5253 \newglossarystyle{superheader}{}%
```

Base it on the `glostylessuper` style:

```
5254 \glossarystyle{super}%
```

Put the glossary in a `supertabular` environment with two columns, a header and no tail:

```
5255 \renewenvironment{theglossary}%
```

```
5256 {\tablehead{\bfseries \entryname & \bfseries \descriptionname\\}%
```

```
5257 \tabletail{}}%
```

```
5258 \begin{supertabular}{lp{\glstdescwidth}}%
```

```
5259 {\end{supertabular}}%
```

```
5260 }
```

**superheaderborder** The superheaderborder style is like the `super` style but with a header and border:

```
5261 \newglossarystyle{superheaderborder}{}%
```

Base it on the `glostylesuper` style:

```
5262 \glossarystyle{super}%
```

Put the glossary in a `supertabular` environment with two columns, a header and horizontal lines above and below the table:

```
5263 \renewenvironment{theglossary}%  
5264   {\tablehead{\hline\bfseries \entryname &  
5265     \bfseries \descriptionname\\ \hline}%  
5266   \tabletail{\hline}  
5267   \begin{supertabular}{|l|p{\glstdescwidth}|}%  
5268   {\end{supertabular}}%  
5269 }
```

`super3col` The `super3col` style is like the `super` style, but with 3 columns:

```
5270 \newglossarystyle{super3col}{%
```

Put the glossary in a `supertabular` environment with three columns and no head or tail:

```
5271 \renewenvironment{theglossary}%  
5272   {\tablehead{}\tabletail{}}%  
5273   \begin{supertabular}{lp{\glstdescwidth}p{\glspagelistwidth}}}%  
5274   {\end{supertabular}}%
```

Do nothing at the start of the table:

```
5275 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
5276 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
5277 \renewcommand*{\glossaryentryfield}[5]{%  
5278   \glstryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\\}%
```

Sub entries on a row (no name, description in second column, page list in last column):

```
5279 \renewcommand*{\glossarysubentryfield}[6]{%  
5280   &  
5281   \glssubentryitem{##2}%  
5282   \glstarget{##2}{\strut}##4 & ##6\\}%
```

Blank row between groups:

```
5283 \renewcommand*{\glsgroupskip}{\ifglsgnpgroupskip\else & & \\fi}%  
5284 }
```

`super3colborder` The `super3colborder` style is like the `super3col` style, but with a border:

```
5285 \newglossarystyle{super3colborder}{%
```

Base it on the `glostylesuper3col` style:

```
5286 \glossarystyle{super3col}%
```

Put the glossary in a supertabular environment with three columns and a horizontal line in the head and tail:

```
5287 \renewenvironment{theglossary}%
5288   {\tablehead{\hline}\tabletail{\hline}%
5289   \begin{supertabular}{|l|p{\glsgdescwidth}|p{\glspagelistwidth}|}%
5290   {\end{supertabular}}%
5291 }
```

**super3colheader** The super3colheader style is like the super3col style but with a header row:

```
5292 \newglossarystyle{super3colheader}{%
```

Base it on the glostylesuper3col style:

```
5293 \glossarystyle{super3col}%
```

Put the glossary in a supertabular environment with three columns, a header and no tail:

```
5294 \renewenvironment{theglossary}%
5295   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
5296   \bfseries\pagelistname\\}\tabletail{}}%
5297   \begin{supertabular}{lp{\glsgdescwidth}p{\glspagelistwidth}}%
5298   {\end{supertabular}}%
5299 }
```

**super3colheaderborder** The super3colheaderborder style is like the super3col style but with a header and border:

```
5300 \newglossarystyle{super3colheaderborder}{%
```

Base it on the glostylesuper3colborder style:

```
5301 \glossarystyle{super3colborder}%
```

Put the glossary in a supertabular environment with three columns, a header with horizontal lines and a horizontal line in the tail:

```
5302 \renewenvironment{theglossary}%
5303   {\tablehead{\hline
5304   \bfseries\entryname&\bfseries\descriptionname&
5305   \bfseries\pagelistname\\ \hline}%
5306   \tabletail{\hline}%
5307   \begin{supertabular}{|l|p{\glsgdescwidth}|p{\glspagelistwidth}|}%
5308   {\end{supertabular}}%
5309 }
```

**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.

```
5310 \newglossarystyle{super4col}{%
```

Put the glossary in a supertabular environment with four columns and no head or tail:

```
5311 \renewenvironment{theglossary}%
5312   {\tablehead{} \tabletail{}}%
5313   \begin{supertabular}{llll}%
5314   \end{supertabular}}%
```

Do nothing at the start of the table:

```
5315 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
5316 \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:

```
5317 \renewcommand*{\glossaryentryfield}[5]{%
```

```
5318 \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:

```
5319 \renewcommand*{\glossarysubentryfield}[6]{%
```

```
5320 &
```

```
5321 \glssubentryitem{##2}%
```

```
5322 \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%
```

Blank row between groups:

```
5323 \renewcommand*{\glsgroupskip}{\ifglsgnoglobskip\else & & \\fi}%
```

```
5324 }
```

**super4colheader** The super4colheader style is like the super4col but with a header row.

```
5325 \newglossarystyle{super4colheader}{}%
```

Base it on the glosstylesuper4col style:

```
5326 \glossarystyle{super4col}%
```

Put the glossary in a supertabular environment with four columns, a header and no tail:

```
5327 \renewenvironment{theglossary}%
```

```
5328 {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
```

```
5329 \bfseries\symbolname &
```

```
5330 \bfseries\pagelistname\\}%
```

```
5331 \tabletail{}}%
```

```
5332 \begin{supertabular}{|l|l|l|l|}%
```

```
5333 {\end{supertabular}}%
```

```
5334 }
```

**super4colborder** The super4colborder style is like the super4col but with a border.

```
5335 \newglossarystyle{super4colborder}{}%
```

Base it on the glosstylesuper4col style:

```
5336 \glossarystyle{super4col}%
```

Put the glossary in a supertabular environment with four columns and a horizontal line in the head and tail:

```
5337 \renewenvironment{theglossary}%
```

```
5338 {\tablehead{\hline}\tabletail{\hline}%
```

```
5339 \begin{supertabular}{|l|l|l|l|}%
```

```
5340 {\end{supertabular}}%
```

```
5341 }
```

`per4colheaderborder` The `super4colheaderborder` style is like the `super4col` but with a header and border.

```
5342 \newglossarystyle{super4colheaderborder}{%
```

Base it on the `glostylesuper4col` style:

```
5343 \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:

```
5344 \renewenvironment{theglossary}%
5345 {\tablehead{\hline\bfseries\entryname&\bfseries\descriptionname&
5346 \bfseries\symbolname &
5347 \bfseries\pagelistname\\\hline}\tabletail{\hline}%
5348 \begin{supertabular}{|l|l|l|l|}}%
5349 {\end{supertabular}}%
5350 }
```

`altsuper4col` The `altsuper4col` glossary style is like `super4col` but has provision for multiline descriptions.

```
5351 \newglossarystyle{altsuper4col}{%
```

Base it on the `glostylesuper4col` style:

```
5352 \glossarystyle{super4col}%
```

Put the glossary in a `supertabular` environment with four columns and no head or tail:

```
5353 \renewenvironment{theglossary}%
5354 {\tablehead{}\tabletail{}}%
5355 \begin{supertabular}{lp{\glsgdescwidth}lp{\glspagelistwidth}}}%
5356 {\end{supertabular}}%
5357 }
```

`altsuper4colheader` The `altsuper4colheader` style is like the `altsuper4col` but with a header row.

```
5358 \newglossarystyle{altsuper4colheader}{%
```

Base it on the `glostylesuper4colheader` style:

```
5359 \glossarystyle{super4colheader}%
```

Put the glossary in a `supertabular` environment with four columns, a header and no tail:

```
5360 \renewenvironment{theglossary}%
5361 {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
5362 \bfseries\symbolname &
5363 \bfseries\pagelistname\\\tabletail{}}%
5364 \begin{supertabular}{lp{\glsgdescwidth}lp{\glspagelistwidth}}}%
5365 {\end{supertabular}}%
5366 }
```

`altsuper4colborder` The `altsuper4colborder` style is like the `altsuper4col` but with a border.

```
5367 \newglossarystyle{altsuper4colborder}{%
```

Base it on the `glostylessuper4colborder` style:

```
5368 \glossarystyle{super4colborder}%
    Put the glossary in a supertabular environment with four columns and a horizontal line in the head and tail:
5369 \renewenvironment{theglossary}%
5370   {\tablehead{\hline}\tabletail{\hline}%
5371    \begin{supertabular}%
5372      {\lllp{\glstdescwidth}\lllp{\glspagelistwidth}}}%
5373   {\end{supertabular}}%
5374 }
```

`per4colheaderborder` The `altsuper4colheaderborder` style is like the `altsuper4col` but with a header and border.

```
5375 \newglossarystyle{altsuper4colheaderborder}{%
    Base it on the glostylessuper4colheaderborder style:
5376 \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:
5377 \renewenvironment{theglossary}%
5378   {\tablehead{\hline
5379     \bfseries\entryname &
5380     \bfseries\descriptionname &
5381     \bfseries\symbolname &
5382     \bfseries\pagelistname\\\hline}%
5383    \tabletail{\hline}%
5384    \begin{supertabular}%
5385      {\lllp{\glstdescwidth}\lllp{\glspagelistwidth}}}%
5386   {\end{supertabular}}%
5387 }
```

### 3.8 Glossary Styles using supertabular environment (glossary-superragged package)

The glossary styles defined in the package use the supertabular environment. These styles are like those provided by the package, except that the multiline columns have ragged right justification.

```
5388 \ProvidesPackage{glossary-superragged}[2012/09/21 v3.03 (NLCT)]
```

Requires the package:

```
5389 \RequirePackage{array}
```

Requires the package:

```
5390 \RequirePackage{supertabular}
```

`\glstdescwidth` This is a length that governs the width of the description column. This may already have been defined.

```

5391 \@ifundefined{glsdescwidth}{%
5392   \newlength{glsdescwidth
5393   \setlength{glsdescwidth}{0.6\hsize}
5394 }{}

```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

```

5395 \@ifundefined{glspagelistwidth}{%
5396   \newlength{glspagelistwidth
5397   \setlength{glspagelistwidth}{0.1\hsize}
5398 }{}

```

`superragged` The `superragged` glossary style uses the `supertabular` environment.

```

5399 \newglossarystyle{superragged}{%

```

Put the glossary in a `supertabular` environment with two columns and no head or tail:

```

5400   \renewenvironment{theglossary}%
5401     {\tablehead{}\tabletail{}}%
5402     \begin{supertabular}{1>{\raggedright}p{glsdescwidth}}}%
5403     {\end{supertabular}}%

```

Do nothing at the start of the table:

```

5404   \renewcommand*{\glossaryheader}{}%

```

No group headings:

```

5405   \renewcommand*{\glsgroupheading}[1]{}%

```

Main (level 0) entries put in a row (name in first column, description and page list in second column):

```

5406   \renewcommand*{\glossaryentryfield}[5]{%
5407     \glstentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%
5408     \tabularnewline}%

```

Sub entries put in a row (no name, description and page list in second column):

```

5409   \renewcommand*{\glossarysubentryfield}[6]{%
5410     &
5411     \glssubentryitem{##2}%
5412     \glstarget{##2}{\strut}##4\glspostdescription\space ##6%
5413     \tabularnewline}%

```

Blank row between groups:

```

5414   \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & \tabularnewline\fi}%
5415 }

```

`superraggedborder` The `superraggedborder` style is like the above, but with horizontal and vertical lines:

```

5416 \newglossarystyle{superraggedborder}{%

```

Base it on the `glostylesuperragged` style:

```

5417   \glossarystyle{superragged}%

```

Put the glossary in a supertabular environment with two columns and a horizontal line in the head and tail:

```
5418 \renewenvironment{theglossary}%
5419   {\tablehead{\hline}\tabletail{\hline}%
5420    \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}|}}%
5421   {\end{supertabular}}%
5422 }
```

**superraggedheader** The superraggedheader style is like the super style, but with a header:

```
5423 \newglossarystyle{superraggedheader}{%
```

Base it on the glostylesuperragged style:

```
5424 \glossarystyle{superragged}%
```

Put the glossary in a supertabular environment with two columns, a header and no tail:

```
5425 \renewenvironment{theglossary}%
5426   {\tablehead{\bfseries \entryname & \bfseries \descriptionname
5427    \tabularnewline}%
5428    \tabletail{}}%
5429   \begin{supertabular}{|l>{\raggedright}p{\glsgdescwidth}}}%
5430   {\end{supertabular}}%
5431 }
```

**superraggedheaderborder** The superraggedheaderborder style is like the superragged style but with a header and border:

```
5432 \newglossarystyle{superraggedheaderborder}{%
```

Base it on the glostylesuper style:

```
5433 \glossarystyle{superragged}%
```

Put the glossary in a supertabular environment with two columns, a header and horizontal lines above and below the table:

```
5434 \renewenvironment{theglossary}%
5435   {\tablehead{\hline\bfseries \entryname &
5436    \bfseries \descriptionname\tabularnewline\hline}%
5437    \tabletail{\hline}
5438    \begin{supertabular}{|l|>{\raggedright}p{\glsgdescwidth}|}}%
5439   {\end{supertabular}}%
5440 }
```

**superragged3col** The superragged3col style is like the superragged style, but with 3 columns:

```
5441 \newglossarystyle{superragged3col}{%
```

Put the glossary in a supertabular environment with three columns and no head or tail:

```
5442 \renewenvironment{theglossary}%
5443   {\tablehead{}\tabletail{}}%
5444   \begin{supertabular}{|l>{\raggedright}p{\glsgdescwidth}%
5445    >{\raggedright}p{\glspagelistwidth}}}%
5446   {\end{supertabular}}%
```

Do nothing at the start of the table:

```
5447 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
5448 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
5449 \renewcommand*{\glossaryentryfield}[5]{%
```

```
5450 \glstryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\tabularnewline}%
```

Sub entries on a row (no name, description in second column, page list in last column):

```
5451 \renewcommand*{\glossarysubentryfield}[6]{%
```

```
5452 &
```

```
5453 \glssubentryitem{##2}%
```

```
5454 \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%
```

Blank row between groups:

```
5455 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & \tabularnewline\fi}%
```

```
5456 }
```

**superragged3colborder** The `superragged3colborder` style is like the `superragged3col` style, but with a border:

```
5457 \newglossarystyle{superragged3colborder}{}%
```

Base it on the `glostylesuperragged3col` style:

```
5458 \glossarystyle{superragged3col}%
```

Put the glossary in a `supertabular` environment with three columns and a horizontal line in the head and tail:

```
5459 \renewenvironment{theglossary}%
```

```
5460 {\tablehead{\hline}\tabletail{\hline}%
```

```
5461 \begin{supertabular}{|l|>{\raggedright}p{\glsdescwidth}|%
```

```
5462 >{\raggedright}p{\glspagelistwidth}|}%
```

```
5463 {\end{supertabular}}%
```

```
5464 }
```

**superragged3colheader** The `superragged3colheader` style is like the `superragged3col` style but with a header row:

```
5465 \newglossarystyle{superragged3colheader}{}%
```

Base it on the `glostylesuperragged3col` style:

```
5466 \glossarystyle{superragged3col}%
```

Put the glossary in a `supertabular` environment with three columns, a header and no tail:

```
5467 \renewenvironment{theglossary}%
```

```
5468 {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
```

```
5469 \bfseries\pagelistname\tabularnewline}\tabletail{}}%
```

```
5470 \begin{supertabular}{l>{\raggedright}p{\glsdescwidth}%
```

```

5471         >{\raggedright}p{\glspagelistwidth}}}%
5472     {\end{supertabular}}}%
5473 }

```

`ght3colheaderborder` The `superragged3colheaderborder` style is like the `superragged3col` style but with a header and border:

```

5474 \newglossarystyle{superragged3colheaderborder}{%

```

Base it on the `glostylesuperragged3colborder` style:

```

5475 \glossarystyle{superragged3colborder}%

```

Put the glossary in a `supertabular` environment with three columns, a header with horizontal lines and a horizontal line in the tail:

```

5476 \renewenvironment{theglossary}%
5477     {\tablehead{\hline
5478         \bfseries\entryname&\bfseries\descriptionname&
5479         \bfseries\pagelistname\tabularnewline\hline}%
5480     \tabletail{\hline}%
5481     \begin{supertabular}{|l|>{\raggedright}p{\glsdescwidth}|%
5482         >{\raggedright}p{\glspagelistwidth}|}%
5483     {\end{supertabular}}}%
5484 }

```

`altsuperragged4col` The `altsuperragged4col` glossary style is like `altsuper4col` style in the package but uses ragged right formatting in the description and page list columns.

```

5485 \newglossarystyle{altsuperragged4col}{%

```

Put the glossary in a `supertabular` environment with four columns and no head or tail:

```

5486 \renewenvironment{theglossary}%
5487     {\tablehead{}\tabletail}%
5488     \begin{supertabular}{|l>{\raggedright}p{\glsdescwidth}l%
5489         >{\raggedright}p{\glspagelistwidth}}}%
5490     {\end{supertabular}}}%

```

Do nothing at the start of the table:

```

5491 \renewcommand*{\glossaryheader}{}%

```

No group headings:

```

5492 \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:

```

5493 \renewcommand*{\glossaryentryfield}[5]{%
5494     \glstryitem{##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:

```

5495 \renewcommand*{\glossarysubentryfield}[6]{%
5496     &
5497     \glssubentryitem{##2}%
5498     \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%

```

Blank row between groups:

```
5499 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & & \tabularnewline\fi}%  
5500 }
```

**altperragged4colheader** The `altperragged4colheader` style is like the `altperragged4col` style but with a header row.

```
5501 \newglossarystyle{altperragged4colheader}{%
```

Base it on the `glostylealtperragged4col` style:

```
5502 \glossarystyle{altperragged4col}%
```

Put the glossary in a `supertabular` environment with four columns, a header and no tail:

```
5503 \renewenvironment{theglossary}%  
5504 {\tablehead{\bfseries\entryname&\bfseries\descriptionname&  
5505 \bfseries\symbolname &  
5506 \bfseries\pagelistname\tabularnewline}\tabletail{}}%  
5507 \begin{supertabular}{l>{\raggedright}p{\glsdescwidth}l%  
5508 >{\raggedright}p{\glspagelistwidth}}}%  
5509 {\end{supertabular}}%  
5510 }
```

**altperragged4colborder** The `altperragged4colborder` style is like the `altperragged4col` style but with a border.

```
5511 \newglossarystyle{altperragged4colborder}{%
```

Base it on the `glostylealtperragged4col` style:

```
5512 \glossarystyle{altperragged4col}%
```

Put the glossary in a `supertabular` environment with four columns and a horizontal line in the head and tail:

```
5513 \renewenvironment{theglossary}%  
5514 {\tablehead{\hline}\tabletail{\hline}%  
5515 \begin{supertabular}%  
5516 {|l|>{\raggedright}p{\glsdescwidth}|l|%  
5517 >{\raggedright}p{\glspagelistwidth}|}%  
5518 {\end{supertabular}}%  
5519 }
```

**altperragged4colheaderborder** The `altperragged4colheaderborder` style is like the `altperragged4col` style but with a header and border.

```
5520 \newglossarystyle{altperragged4colheaderborder}{%
```

Base it on the `glostylealtperragged4col` style:

```
5521 \glossarystyle{altperragged4col}%
```

Put the glossary in a `supertabular` environment with four columns and a header bordered by horizontal lines and a horizontal line in the tail:

```
5522 \renewenvironment{theglossary}%  
5523 {\tablehead{\hline
```

```

5524      \bfseries\entryname &
5525      \bfseries\descriptionname &
5526      \bfseries\symbolname &
5527      \bfseries\pagelistname\tabularnewline\hline}%
5528      \tabletail{\hline}%
5529      \begin{supertabular}%
5530          {\l|>{\raggedright}p{\glsdescwidth}|l|}%
5531          >{\raggedright}p{\glspagelistwidth}|}}%
5532      {\end{supertabular}}%
5533 }

```

### 3.9 Tree Styles (glossary-tree.sty)

The style file defines glossary styles that have a tree-like structure. These are designed for hierarchical glossaries.

```

5534 \ProvidesPackage{glossary-tree}[2012/09/21 v3.03 (NLCT)]

```

**index** The index glossary style is similar in style to the way indices are usually typeset using `\item`, `\subitem` and `\subsubitem`. The entry name is set in bold. If an entry has a symbol, it is placed in brackets after the name. Then the description is displayed, followed by the number list. This style allows up to three levels.

```

5535 \newglossarystyle{index}{%

```

Set the paragraph indentation and skip and define `\item` to be the same as that used by the index:

```

5536   \renewenvironment{theglossary}%
5537   {\setlength{\parindent}{0pt}%
5538   \setlength{\parskip}{0pt plus 0.3pt}%
5539   \let\item\@idxitem}%
5540   {}%

```

Do nothing at the start of the environment:

```

5541   \renewcommand*{\glossaryheader}{}%

```

No group headers:

```

5542   \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.

```

5543 \renewcommand*{\glossaryentryfield}[5]{%
5544 \item\glstentryitem{##1}\textbf{\glstarget{##1}{##2}}%
5545 \ifx\relax##4\relax
5546   \else
5547     \space{##4}%
5548   \fi
5549   \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.

```

5550 \renewcommand*{\glossarysubentryfield}[6]{%
5551   \ifcase##1\relax
5552     % level 0
5553     \item
5554   \or
5555     % level 1
5556     \subitem
5557     \glssubentryitem{##2}%
5558   \else
5559     % all other levels
5560     \subsubitem
5561   \fi
5562   \textbf{\glstarget{##2}{##3}}%
5563   \ifx\relax##5\relax
5564   \else
5565     \space{##5}%
5566   \fi
5567   \space##4\glspostdescription\space ##6}%

```

Vertical gap between groups is the same as that used by indices:

```

5568 \renewcommand*{\glsgroupskip}{\ifglsgnোগroupskip\else\indexspace\fi}}

```

**indexgroup** The `indexgroup` style is like the `index` style but has headings.

```

5569 \newglossarystyle{indexgroup}{%

```

Base it on the `glostyleindex` style:

```

5570 \glossarystyle{index}%

```

Add a heading for each group. This puts the group's title in bold followed by a vertical gap.

```

5571 \renewcommand*{\glsgroupheading}[1]{%
5572   \item\textbf{\glsgroupgetgrouptitle{##1}}\indexspace}%
5573 }

```

**indexhypergroup** The `indexhypergroup` style is like the `indexgroup` style but has hyper navigation.

```

5574 \newglossarystyle{indexhypergroup}{%

```

Base it on the `glostyleindex` style:

```

5575 \glossarystyle{index}%

```

Put navigation links to the groups at the start of the glossary:

```

5576 \renewcommand*{\glossaryheader}{%
5577   \item\textbf{\glshnavigation}\indexspace}%

```

Add a heading for each group (with a target). The group's title is in bold followed by a vertical gap.

```

5578 \renewcommand*{\glsgroupheading}[1]{%
5579   \item\textbf{\glshnavhypertarget{##1}{\glsgroupgetgrouptitle{##1}}}%
5580   \indexspace}%
5581 }

```

**tree** The tree glossary style is similar in style to the index style, but can have arbitrary levels.

```
5582 \newglossarystyle{tree}{%
```

Set the paragraph indentation and skip:

```
5583 \renewenvironment{theglossary}%
5584 {\setlength{\parindent}{0pt}%
5585 \setlength{\parskip}{0pt plus 0.3pt}}%
5586 {}%
```

Do nothing at the start of the theglossary environment:

```
5587 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
5588 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries: name in bold, followed by symbol in brackets (if it exists), the description and the page list:

```
5589 \renewcommand{\glossaryentryfield}[5]{%
5590 \hangindent0pt\relax
5591 \parindent0pt\relax
5592 \glstentryitem{##1}\textbf{\glstarget{##1}{##2}}%
5593 \ifx\relax##4\relax
5594 \else
5595 \space{##4}%
5596 \fi
5597 \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.

```
5598 \renewcommand{\glossarysubentryfield}[6]{%
5599 \hangindent##1\glstreeindent\relax
5600 \parindent##1\glstreeindent\relax
5601 \ifnum##1=1\relax
5602 \glssubentryitem{##2}%
5603 \fi
5604 \textbf{\glstarget{##2}{##3}}%
5605 \ifx\relax##5\relax
5606 \else
5607 \space{##5}%
5608 \fi
5609 \space##4\glspostdescription\space ##6\par}%
```

Vertical gap between groups is the same as that used by indices:

```
5610 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}}
```

**treegroup** Like the tree style but the glossary groups have headings.

```
5611 \newglossarystyle{treegroup}{%
```

Base it on the glstyletree style:

```
5612 \glossarystyle{tree}%
```

Each group has a heading (in bold) followed by a vertical gap):

```
5613 \renewcommand{\glsgroupheading}[1]{\par
5614 \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
5615 }
```

**treehypergroup** The treehypergroup style is like the treegroup style, but has a set of links to the groups at the start of the glossary.

```
5616 \newglossarystyle{treehypergroup}{%
```

Base it on the glostyletree style:

```
5617 \glossarystyle{tree}%
```

Put navigation links to the groups at the start of the theglossary environment:

```
5618 \renewcommand*{\glossaryheader}{%
5619 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
5620 \renewcommand*{\glsgroupheading}[1]{%
5621 \par\noindent
5622 \textbf{\glsnavigationtarget{##1}{\glsgrouptitle{##1}}}\par
5623 \indexspace}%
5624 }
```

**\glstreeindent** Length governing left indent for each level of the tree style.

```
5625 \newlength\glstreeindent
5626 \setlength{\glstreeindent}{10pt}
```

**treenoname** The treenoname glossary style is like the tree style, but doesn't print the name or symbol for sub-levels.

```
5627 \newglossarystyle{treenoname}{%
```

Set the paragraph indentation and skip:

```
5628 \renewenvironment{theglossary}%
5629 {\setlength{\parindent}{0pt}%
5630 \setlength{\parskip}{0pt plus 0.3pt}}%
5631 {}%
```

No header:

```
5632 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
5633 \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.

```
5634 \renewcommand{\glossaryentryfield}[5]{%
5635 \hangindent0pt\relax
5636 \parindent0pt\relax
5637 \glstentryitem{##1}\textbf{\glstarget{##1}{##2}}%
5638 \ifx\relax##4\relax
5639 \else
```

```

5640     \space{##4}%
5641     \fi
5642     \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.

```

5643 \renewcommand{\glossarysubentryfield}[6]{%
5644     \hangindent##1\glstreeindent\relax
5645     \parindent##1\glstreeindent\relax
5646     \ifnum##1=1\relax
5647         \glssubentryitem{##2}%
5648     \fi
5649     \glstarget{##2}{\strut}%
5650     ##4\glspostdescription\space ##6\par}%

```

Vertical gap between groups is the same as that used by indices:

```

5651 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
5652 }

```

**treenonamegroup** Like the `treenoname` style but the glossary groups have headings.

```

5653 \newglossarystyle{treenonamegroup}{%

```

Base it on the `glostyletreenoname` style:

```

5654 \glossarystyle{treenoname}%

```

Give each group a heading:

```

5655 \renewcommand{\glsgroupheading}[1]{\par
5656     \noindent\textbf{\glsgrouptitle{##1}}\par\indexspace}%
5657 }

```

**treenonamehypergroup** The `treenonamehypergroup` style is like the `treenonamegroup` style, but has a set of links to the groups at the start of the glossary.

```

5658 \newglossarystyle{treenonamehypergroup}{%

```

Base it on the `glostyletreenoname` style:

```

5659 \glossarystyle{treenoname}%

```

Put navigation links to the groups at the start of the `theglossary` environment:

```

5660 \renewcommand*{\glossaryheader}{%
5661     \par\noindent\textbf{\glsnavigation}\par\indexspace}%

```

Each group has a heading (in bold with a target) followed by a vertical gap):

```

5662 \renewcommand*{\glsgroupheading}[1]{%
5663     \par\noindent
5664     \textbf{\glsnavigationhypertarget{##1}{\glsgrouptitle{##1}}}\par
5665     \indexspace}%
5666 }

```

**\glsssetwidest** `\glsssetwidest[ $\langle level \rangle$ ]{ $\langle text \rangle$ }` sets the widest text for the given level. It is used by the `alttree` glossary styles to determine the indentation of each level.

```

5667 \newcommand*{\glsssetwidest}[2][0]{%

```

```

5668 \expandafter\def\csname @glswidestname\romannumeral#1\endcsname{%
5669 #2}%
5670 }

```

\@glswidestname Initialise \@glswidestname.

```

5671 \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.

```

5672 \newglossarystyle{alttree}{%

```

Redefine theglossary environment.

```

5673 \renewenvironment{theglossary}%
5674 {\def\@gls@prevlevel{-1}%
5675 \mbox{}\par}%
5676 {\par}%

```

Set the header and group headers to nothing.

```

5677 \renewcommand*{\glossaryheader}{}%
5678 \renewcommand*{\glsgroupheading}[1]{}%

```

Redefine the way that the level 0 entries are displayed.

```

5679 \renewcommand{\glossaryentryfield}[5]{%

```

If the level hasn't changed, keep the same settings, otherwise change \glstreeindent accordingly.

```

5680 \ifnum\@gls@prevlevel=0\relax
5681 \else

```

Find out how big the indentation should be by measuring the widest entry.

```

5682 \settowidth{\glstreeindent}{\textbf{\@glswidestname\space}}%

```

Set the hangindent and paragraph indent.

```

5683 \hangindent\glstreeindent
5684 \parindent\glstreeindent
5685 \fi

```

Put the name to the left of the paragraph block.

```

5686 \makebox[0pt][r]{\makebox[\glstreeindent][l]{%
5687 \glsentryitem{##1}\textbf{\glstarget{##1}{##2}}}%

```

If the symbol is missing, ignore it, otherwise put it in brackets.

```

5688 \ifx\relax##4\relax
5689 \else
5690 (##4)\space
5691 \fi

```

Do the description followed by the description terminator and location list.

```

5692 ##3\glspostdescription \space ##5\par

```

Set the previous level to 0.

```
5693 \def\@gls@prevlevel{0}%  
5694 }%
```

Redefine the way sub-entries are displayed.

```
5695 \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.

```
5696 \ifnum##1=1\relax  
5697 \glssubentryitem{##2}%  
5698 \fi
```

If the level hasn't changed, keep the same settings, otherwise adjust `\glstreeindent` accordingly.

```
5699 \ifnum\@gls@prevlevel=##1\relax  
5700 \else
```

Compute the widest entry for this level, or for level 0 if not defined for this level.

Store in `\gls@tmplen`

```
5701 \@ifundefined{@glswidestname\romannumeral##1}{%  
5702 \settowidth{\gls@tmplen}{\textbf{@glswidestname\space}}}%  
5703 \settowidth{\gls@tmplen}{\textbf{%  
5704 \csname @glswidestname\romannumeral##1\endcsname\space}}}%
```

Determine if going up or down a level

```
5705 \ifnum\@gls@prevlevel<##1\relax
```

Depth has increased, so add the width of the widest entry to `\glstreeindent`.

```
5706 \setlength\glstreeindent\gls@tmplen  
5707 \addtolength\glstreeindent\parindent  
5708 \parindent\glstreeindent  
5709 \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`.

```
5710 \@ifundefined{@glswidestname\romannumeral\@gls@prevlevel}{%  
5711 \settowidth{\glstreeindent}{\textbf{%  
5712 @glswidestname\space}}}%  
5713 \settowidth{\glstreeindent}{\textbf{%  
5714 \csname @glswidestname\romannumeral\@gls@prevlevel  
5715 \endcsname\space}}}%
```

Subtract this length from the previous level's paragraph indent and set to `\glstreeindent`.

```
5716 \addtolength\parindent{-\glstreeindent}%  
5717 \setlength\glstreeindent\parindent  
5718 \fi  
5719 \fi
```

Set the hanging indentation.

```
5720 \hangindent\glstreeindent
```

Put the name to the left of the paragraph block

```
5721 \makebox[0pt][r]{\makebox[\gls@tmplen][l]{%  
5722 \textbf{\glstarget{##2}{##3}}}%
```

If the symbol is missing, ignore it, otherwise put it in brackets.

```
5723 \ifx##5\relax\relax  
5724 \else  
5725 (##5)\space  
5726 \fi
```

Do the description followed by the description terminator and location list.

```
5727 ##4\glspostdescription\space ##6\par
```

Set the previous level macro to the current level.

```
5728 \def\@gls@prevlevel{##1}%  
5729 }%
```

Vertical gap between groups is the same as that used by indices:

```
5730 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%  
5731 }
```

**alttreegroup** Like the alttree style but the glossary groups have headings.

```
5732 \newglossarystyle{alttreegroup}{%
```

Base it on the glostylealttree style:

```
5733 \glossarystyle{alttree}%
```

Give each group a heading.

```
5734 \renewcommand{\glsgroupheading}[1]{\par  
5735 \def\@gls@prevlevel{-1}%  
5736 \hangindent0pt\relax  
5737 \parindent0pt\relax  
5738 \textbf{\glsgetgrouptitle{##1}}\par\indexspace}%  
5739 }
```

**alttreehypergroup** The alttreehypergroup style is like the alttreegroup style, but has a set of links to the groups at the start of the glossary.

```
5740 \newglossarystyle{alttreehypergroup}{%
```

Base it on the glostylealttree style:

```
5741 \glossarystyle{alttree}%
```

Put the navigation links in the header

```
5742 \renewcommand*{\glossaryheader}{%  
5743 \par  
5744 \def\@gls@prevlevel{-1}%  
5745 \hangindent0pt\relax  
5746 \parindent0pt\relax  
5747 \textbf{\glsnavigation}\par\indexspace}%
```

Put a hypertarget at the start of each group

```

5748 \renewcommand*\glsgroupheading}[1]{%
5749   \par
5750   \def\@gls@prevlevel{-1}%
5751   \hangindent0pt\relax
5752   \parindent0pt\relax
5753   \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
5754   \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.

```

5755 \NeedsTeXFormat{LaTeX2e}
5756 \ProvidesPackage{glossaries-compatible-207}[2011/04/02 v1.0 (NLCT)]

```

`\GlsAddXdyAttribute` Adds an attribute in old format.

```

5757 \ifglsxindy
5758   \renewcommand*\GlsAddXdyAttribute[1]{%
5759     \edef\@xdyattributes{\@xdyattributes ^^J \string"#1\string"}%
5760     \expandafter\toks@\expandafter{\@xdylocref}%
5761     \edef\@xdylocref{\the\toks@ ^^J%
5762 (markup-locref
5763 :open \string"\string~\string\setentrycounter
5764   {\noexpand\glscounter}%
5765   \expandafter\string\csname#1\endcsname
5766   \expandafter\@gobble\string\{\string" ^^J
5767 :close \string"\expandafter\@gobble\string\}\string" ^^J
5768 :attr \string"#1\string"))}

```

Only has an effect before `\writeist`:

```

5769 \fi

```

`\GlsAddXdyCounters`

```

5770 \renewcommand*\GlsAddXdyCounters[1]{%
5771   \GlossariesWarning{\string\GlsAddXdyCounters\space not available
5772     in compatibility mode.}%
5773 }

```

Add predefined attributes

```

5774 \GlsAddXdyAttribute{glsnumberformat}
5775 \GlsAddXdyAttribute{textrm}
5776 \GlsAddXdyAttribute{textsf}
5777 \GlsAddXdyAttribute{texttt}
5778 \GlsAddXdyAttribute{textbf}
5779 \GlsAddXdyAttribute{textmd}
5780 \GlsAddXdyAttribute{textit}

```

```

5781 \GlsAddXdyAttribute{textup}
5782 \GlsAddXdyAttribute{textsl}
5783 \GlsAddXdyAttribute{textsc}
5784 \GlsAddXdyAttribute{emph}
5785 \GlsAddXdyAttribute{glshypernumber}
5786 \GlsAddXdyAttribute{hyperrrm}
5787 \GlsAddXdyAttribute{hypersf}
5788 \GlsAddXdyAttribute{hypertt}
5789 \GlsAddXdyAttribute{hyperbf}
5790 \GlsAddXdyAttribute{hypermd}
5791 \GlsAddXdyAttribute{hyperit}
5792 \GlsAddXdyAttribute{hyperup}
5793 \GlsAddXdyAttribute{hypersl}
5794 \GlsAddXdyAttribute{hypersc}
5795 \GlsAddXdyAttribute{hyperemph}

```

\GlsAddXdyLocation Restore v2.07 definition:

```

5796 \ifglxindy
5797 \renewcommand*{\GlsAddXdyLocation}[2]{%
5798 \edef\@xdyuserlocationdefs{%
5799 \@xdyuserlocationdefs ^^J%
5800 (define-location-class \string"#1\string"^^J\space\space
5801 \space(#2))
5802 }%
5803 \edef\@xdyuserlocationnames{%
5804 \@xdyuserlocationnames^^J\space\space\space
5805 \string"#1\string"}%
5806 }
5807 \fi

```

\@do@wrglossary

```

5808 \renewcommand{\@do@wrglossary}[1]{%
  Determine whether to use xindy or makeindex syntax
5809 \ifglxindy
  Need to determine if the formatting information starts with a ( or ) indicating a
  range.
5810 \expandafter\@glo@check@mkidxrangechar\@glsnumberformat\@nil
5811 \def\@glo@range{}%
5812 \expandafter\if\@glo@prefix(\relax
5813 \def\@glo@range{:open-range}%
5814 \else
5815 \expandafter\if\@glo@prefix)\relax
5816 \def\@glo@range{:close-range}%
5817 \fi
5818 \fi
  Get the location and escape any special characters
5819 \protected@edef\@glslocref{\theglstrycounter}%
5820 \@gls@checkmkidxchars\@glslocref

```

Write to the glossary file using xindy syntax.

```
5821 \glossary[\csname glo@#1@type\endcsname]{%
5822 (indexentry :tkey (\csname glo@#1@index\endcsname)
5823 :locoref \string"\@glslocoref\string" %
5824 :attr \string"\@glo@suffix\string" \@glo@range
5825 )
5826 }%
5827 \else
```

Convert the format information into the format required for makeindex

```
5828 \@set@glo@numformat\@glo@numfmt\@gls@counter\@glsnumberformat
```

Write to the glossary file using makeindex syntax.

```
5829 \glossary[\csname glo@#1@type\endcsname]{%
5830 \string\glossaryentry{\csname glo@#1@index\endcsname
5831 \@gls@encapchar\@glo@numfmt}{\theglsentrycounter}}%
5832 \fi
5833 }
```

\@set@glo@numformat Only had 3 arguments in v2.07

```
5834 \def\@set@glo@numformat#1#2#3{%
5835 \expandafter\@glo@check@mkidxrangechar#3\@nil
5836 \protected@edef#1{%
5837 \@glo@prefix setentrycounter[] {#2}%
5838 \expandafter\string\csname\@glo@suffix\endcsname
5839 }%
5840 \@gls@checkmkidxchars#1%
5841 }
```

\writeist Redefine \writeist back to the way it was in v2.07, but change \istfile to \glswrite.

```
5842 \ifglsxindy
5843 \def\writeist{%
5844 \openout\glswrite=\istfilename
5845 \write\glswrite{;; xindy style file created by the glossaries
5846 package in compatible-2.07 mode}%
5847 \write\glswrite{;; for document '\jobname' on
5848 \the\year-\the\month-\the\day}%
5849 \write\glswrite{^^J; required styles^^J}
5850 \@for\@xdystyle:=\@xdyrequiredstyles\do{%
5851 \ifx\@xdystyle\@empty
5852 \else
5853 \protected@write\glswrite{{(require
5854 \string"\@xdystyle.xdy\string")}}%
5855 \fi
5856 }%
5857 \write\glswrite{^^J%
5858 ; list of allowed attributes (number formats)^^J}%
5859 \write\glswrite{(define-attributes ((\@xdyattributes)))}%
5860 \write\glswrite{^^J; user defined alphabets^^J}%
```

```

5861 \write\glswrite{\@xdyuseralphabets}%
5862 \write\glswrite{^^J; location class definitions^^J}%
5863 \protected@edef\@gls@roman{\@roman{0}\string"
5864 \string"roman-numbers-lowercase\string" :sep \string"}}%
5865 \@onelevel@sanitize\@gls@roman
5866 \edef\@tmp{\string" \string"roman-numbers-lowercase\string"
5867 :sep \string"}%
5868 \@onelevel@sanitize\@tmp
5869 \ifx\@tmp\@gls@roman
5870 \write\glswrite{(define-location-class
5871 \string"roman-page-numbers\string"^^J\space\space\space
5872 (\string"roman-numbers-lowercase\string")
5873 :min-range-length \@glsminrange)}}%
5874 \else
5875 \write\glswrite{(define-location-class
5876 \string"roman-page-numbers\string"^^J\space\space\space
5877 (:sep "\@gls@roman")
5878 :min-range-length \@glsminrange)}}%
5879 \fi
5880 \write\glswrite{(define-location-class
5881 \string"Roman-page-numbers\string"^^J\space\space\space
5882 (\string"roman-numbers-uppercase\string")
5883 :min-range-length \@glsminrange)}}%
5884 \write\glswrite{(define-location-class
5885 \string"arabic-page-numbers\string"^^J\space\space\space
5886 (\string"arabic-numbers\string")
5887 :min-range-length \@glsminrange)}}%
5888 \write\glswrite{(define-location-class
5889 \string"alpha-page-numbers\string"^^J\space\space\space
5890 (\string"alpha\string")
5891 :min-range-length \@glsminrange)}}%
5892 \write\glswrite{(define-location-class
5893 \string"Alpha-page-numbers\string"^^J\space\space\space
5894 (\string"ALPHA\string")
5895 :min-range-length \@glsminrange)}}%
5896 \write\glswrite{(define-location-class
5897 \string"Appendix-page-numbers\string"^^J\space\space\space
5898 (\string"ALPHA\string"
5899 :sep \string"\@glsAlphacompositor\string"
5900 \string"arabic-numbers\string")
5901 :min-range-length \@glsminrange)}}%
5902 \write\glswrite{(define-location-class
5903 \string"arabic-section-numbers\string"^^J\space\space\space
5904 (\string"arabic-numbers\string"
5905 :sep \string"\glscompositor\string"
5906 \string"arabic-numbers\string")
5907 :min-range-length \@glsminrange)}}%
5908 \write\glswrite{^^J; user defined location classes}%
5909 \write\glswrite{\@xdyuserlocationdefs}%

```

```

5910 \write\glswrite{^^J; define cross-reference class^^J}%
5911 \write\glswrite{(define-crossref-class \string"see\string"
5912   :unverified )}%
5913 \write\glswrite{(markup-crossref-list
5914   :class \string"see\string"^^J\space\space\space
5915   :open \string"\string\glseeformat\string"
5916   :close \string"{}\string")}%
5917 \write\glswrite{^^J; define the order of the location classes}%
5918 \write\glswrite{(define-location-class-order
5919   (\@xdylocationclassorder))}%
5920 \write\glswrite{^^J; define the glossary markup^^J}%
5921 \write\glswrite{(markup-index^^J\space\space\space
5922   :open \string"\string
5923     \glossarysection[\string\glossarytoctitle]{\string
5924     \glossarytitle}\string\glossarypreamble\string~n\string\begin
5925     {theglossary}\string\glossaryheader\string~n\string" ^^J\space
5926     \space\space:close \string"\expandafter\@gobble
5927     \string%\string~n\string
5928     \end{theglossary}\string\glossarypostamble
5929     \string~n\string" ^^J\space\space\space
5930   :tree)}}%
5931 \write\glswrite{(markup-letter-group-list
5932   :sep \string"\string\glsgroupskip\string~n\string")}%
5933 \write\glswrite{(markup-indexentry
5934   :open \string"\string\relax \string\glresetentrylist
5935     \string~n\string")}%
5936 \write\glswrite{(markup-locclass-list :open
5937   \string"\glsoopenbrace\string\glossaryentrynumbers
5938     \glsoopenbrace\string\relax\space \string"^^J\space\space\space
5939   :sep \string", \string"
5940   :close \string"\glsclosebrace\glsclosebrace\string")}%
5941 \write\glswrite{(markup-locref-list
5942   :sep \string"\string\delimN\space\string")}%
5943 \write\glswrite{(markup-range
5944   :sep \string"\string\delimR\space\string")}%
5945 \@onelevel@sanitize\gls@suffiF
5946 \@onelevel@sanitize\gls@suffiFF
5947 \ifx\gls@suffiF\@empty
5948 \else
5949   \write\glswrite{(markup-range
5950     :close "\gls@suffiF" :length 1 :ignore-end)}}%
5951 \fi
5952 \ifx\gls@suffiFF\@empty
5953 \else
5954   \write\glswrite{(markup-range
5955     :close "\gls@suffiFF" :length 2 :ignore-end)}}%
5956 \fi
5957 \write\glswrite{^^J; define format to use for locations^^J}%
5958 \write\glswrite{\@xdylocref}%

```

```

5959 \write\glswrite{^^J; define letter group list format^^J}%
5960 \write\glswrite{(markup-letter-group-list
5961   :sep \string"\string\glsgroupskip\string~n\string")}%
5962 \write\glswrite{^^J; letter group headings^^J}%
5963 \write\glswrite{(markup-letter-group
5964   :open-head \string"\string\glsgroupheading
5965   \glsoopenbrace\string"^^J\space\space\space
5966   :close-head \string"\glsclosebrace\string")}%
5967 \write\glswrite{^^J; additional letter groups^^J}%
5968 \write\glswrite{\@xdylettergroups}%
5969 \write\glswrite{^^J; additional sort rules^^J}
5970 \write\glswrite{\@xdysortrules}%
5971 \noist}
5972 \else
5973 \edef\@gls@actualchar{\string?}
5974 \edef\@gls@encapchar{\string|}
5975 \edef\@gls@levelchar{\string!}
5976 \edef\@gls@quotechar{\string"}
5977 \def\writeist{\relax
5978   \openout\glswrite=\istfilename
5979   \write\glswrite{\expandafter\@gobble\string\% makeindex style file
5980     created by the glossaries package}
5981   \write\glswrite{\expandafter\@gobble\string\% for document
5982     '\jobname' on \the\year-\the\month-\the\day}
5983   \write\glswrite{actual '\@gls@actualchar'}
5984   \write\glswrite{encap '\@gls@encapchar'}
5985   \write\glswrite{level '\@gls@levelchar'}
5986   \write\glswrite{quote '\@gls@quotechar'}
5987   \write\glswrite{keyword \string"\string\glossaryentry\string"}
5988   \write\glswrite{preamble \string"\string\glossarysection[\string
5989     \glossarytoctitle]{\string\glossarytitle}\string
5990     \glossarypreamble\string\n\string\begin{theglossary}\string
5991     \glossaryheader\string\n\string"}
5992   \write\glswrite{postamble \string"\string\%\string\n\string
5993     \end{theglossary}\string\glossarypostamble\string\n
5994     \string"}
5995   \write\glswrite{group_skip \string"\string\glsgroupskip\string\n
5996     \string"}
5997   \write\glswrite{item_0 \string"\string%\string\n\string"}
5998   \write\glswrite{item_1 \string"\string%\string\n\string"}
5999   \write\glswrite{item_2 \string"\string%\string\n\string"}
6000   \write\glswrite{item_01 \string"\string%\string\n\string"}
6001   \write\glswrite{item_x1
6002     \string"\string\relax \string\glsresetentrylist\string\n
6003     \string"}
6004   \write\glswrite{item_12 \string"\string%\string\n\string"}
6005   \write\glswrite{item_x2
6006     \string"\string\relax \string\glsresetentrylist\string\n
6007     \string"}

```

```

6008 \write\glswrite{delim_0 \string"\string\{\string
6009 \glossaryentrynumbers\string\{\string\relax \string"}
6010 \write\glswrite{delim_1 \string"\string\{\string
6011 \glossaryentrynumbers\string\{\string\relax \string"}
6012 \write\glswrite{delim_2 \string"\string\{\string
6013 \glossaryentrynumbers\string\{\string\relax \string"}
6014 \write\glswrite{delim_t \string"\string\}\string\}\string"}
6015 \write\glswrite{delim_n \string"\string\delimN \string"}
6016 \write\glswrite{delim_r \string"\string\delimR \string"}
6017 \write\glswrite{headings_flag 1}
6018 \write\glswrite{heading_prefix
6019 \string"\string\glsgroupheading\string\{\string"}
6020 \write\glswrite{heading_suffix
6021 \string"\string\}\string\relax
6022 \string\glsresetentrylist \string"}
6023 \write\glswrite{symhead_positive \string"glssymbols\string"}
6024 \write\glswrite{numhead_positive \string"glssymbols\string"}
6025 \write\glswrite{page_compositor \string"glscpositor\string"}
6026 \@gls@escbsdq\gls@suffixF
6027 \@gls@escbsdq\gls@suffixFF
6028 \ifx\gls@suffixF\@empty
6029 \else
6030 \write\glswrite{suffix_2p \string"\gls@suffixF\string"}
6031 \fi
6032 \ifx\gls@suffixFF\@empty
6033 \else
6034 \write\glswrite{suffix_3p \string"\gls@suffixFF\string"}
6035 \fi
6036 \noist
6037 }
6038 \fi

\noist
6039 \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.

```
6040 \NeedsTeXFormat{LaTeX2e}
```

Package version number now in line with main glossaries package number but will only be updated when glossaries-accsupp.sty is modified.

```
6041 \ProvidesPackage{glossaries-accsupp}[2011/04/02 v3.0 (NLCT)]
```

```
6042 Experimental glossaries accessibility]
```

Pass all options to glossaries:

```
6043 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{glossaries}}
```

Process options:

```
6044 \ProcessOptions
```

Required packages:

```
6045 \RequirePackage{glossaries}
```

```
6046 \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:

```
6047 \define@key{glossentry}{access}{%
6048   \def\@glo@access{#1}%
6049 }
```

**textaccess** The replacement text corresponding to the text key:

```
6050 \define@key{glossentry}{textaccess}{%
6051   \def\@glo@textaccess{#1}%
6052 }
```

**firstaccess** The replacement text corresponding to the first key:

```
6053 \define@key{glossentry}{firstaccess}{%
6054   \def\@glo@firstaccess{#1}%
6055 }
```

**pluralaccess** The replacement text corresponding to the plural key:

```
6056 \define@key{glossentry}{pluralaccess}{%
6057   \def\@glo@pluralaccess{#1}%
6058 }
```

**firstpluralaccess** The replacement text corresponding to the firstplural key:

```
6059 \define@key{glossentry}{firstpluralaccess}{%
6060   \def\@glo@firstpluralaccess{#1}%
6061 }
```

**symbolaccess** The replacement text corresponding to the symbol key:

```
6062 \define@key{glossentry}{symbolaccess}{%
6063   \def\@glo@symbolaccess{#1}%
6064 }
```

**symbolpluralaccess** The replacement text corresponding to the symbolplural key:

```
6065 \define@key{glossentry}{symbolpluralaccess}{%
6066   \def\@glo@symbolpluralaccess{#1}%
6067 }
```

descriptionaccess The replacement text corresponding to the description key:

```

6068 \define@key{glossentry}{descriptionaccess}{%
6069   \def\@glo@descaccess{#1}%
6070 }

```

descriptionpluralaccess The replacement text corresponding to the descriptionplural key:

```

6071 \define@key{glossentry}{descriptionpluralaccess}{%
6072   \def\@glo@descpluralaccess{#1}%
6073 }

```

shortaccess The replacement text corresponding to the short key:

```

6074 \define@key{glossentry}{shortaccess}{%
6075   \def\@glo@shortaccess{#1}%
6076 }

```

shortpluralaccess The replacement text corresponding to the shortplural key:

```

6077 \define@key{glossentry}{shortpluralaccess}{%
6078   \def\@glo@shortpluralaccess{#1}%
6079 }

```

longaccess The replacement text corresponding to the long key:

```

6080 \define@key{glossentry}{longaccess}{%
6081   \def\@glo@longaccess{#1}%
6082 }

```

longpluralaccess The replacement text corresponding to the longplural key:

```

6083 \define@key{glossentry}{longpluralaccess}{%
6084   \def\@glo@longpluralaccess{#1}%
6085 }

```

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={\glsaccsup{inches}{in}}.

\@gls@noaccess Indicates that no replacement text has been provided.

```

6086 \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):

```

6087 \let\@gls@oldnewglossaryentryprehook\@newglossaryentryprehook
6088 \renewcommand*{\@newglossaryentryprehook}{%
6089   \@gls@oldnewglossaryentryprehook
6090   \def\@glo@access{\@glo@symbol}%

```

Initialise the other keys:

```

6091   \def\@glo@textaccess{\@glo@access}%
6092   \def\@glo@firstaccess{\@glo@access}%
6093   \def\@glo@pluralaccess{\@glo@textaccess}%
6094   \def\@glo@firstpluralaccess{\@glo@pluralaccess}%

```

```

6095 \def\@glo@symbolaccess{\relax}%
6096 \def\@glo@symbolpluralaccess{\@glo@symbolaccess}%
6097 \def\@glo@descaccess{\relax}%
6098 \def\@glo@descpluralaccess{\@glo@descaccess}%
6099 \def\@glo@shortaccess{\relax}%
6100 \def\@glo@shortpluralaccess{\@glo@shortaccess}%
6101 \def\@glo@longaccess{\relax}%
6102 \def\@glo@longpluralaccess{\@glo@longaccess}%
6103 }

```

Add to the end hook:

```

6104 \let\@gls@oldnewglossaryentryposthook\@newglossaryentryposthook
6105 \renewcommand*{\@newglossaryentryposthook}{%
6106 \@gls@oldnewglossaryentryposthook

```

Store the access information:

```

6107 \expandafter
6108 \protected@xdef\csname glo@\@glo@label @access\endcsname{%
6109 \@glo@access}%
6110 \expandafter
6111 \protected@xdef\csname glo@\@glo@label @textaccess\endcsname{%
6112 \@glo@textaccess}%
6113 \expandafter
6114 \protected@xdef\csname glo@\@glo@label @firstaccess\endcsname{%
6115 \@glo@firstaccess}%
6116 \expandafter
6117 \protected@xdef\csname glo@\@glo@label @pluralaccess\endcsname{%
6118 \@glo@pluralaccess}%
6119 \expandafter
6120 \protected@xdef\csname glo@\@glo@label @firstpluralaccess\endcsname{%
6121 \@glo@firstpluralaccess}%
6122 \expandafter
6123 \protected@xdef\csname glo@\@glo@label @symbolaccess\endcsname{%
6124 \@glo@symbolaccess}%
6125 \expandafter
6126 \protected@xdef\csname glo@\@glo@label @symbolpluralaccess\endcsname{%
6127 \@glo@symbolpluralaccess}%
6128 \expandafter
6129 \protected@xdef\csname glo@\@glo@label @descaccess\endcsname{%
6130 \@glo@descaccess}%
6131 \expandafter
6132 \protected@xdef\csname glo@\@glo@label @descpluralaccess\endcsname{%
6133 \@glo@descpluralaccess}%
6134 \expandafter
6135 \protected@xdef\csname glo@\@glo@label @shortaccess\endcsname{%
6136 \@glo@shortaccess}%
6137 \expandafter
6138 \protected@xdef\csname glo@\@glo@label @shortpluralaccess\endcsname{%
6139 \@glo@shortpluralaccess}%
6140 \expandafter

```

```

6141 \protected@xdef\csname glo@\@glo@label @longaccess\endcsname{%
6142 \@glo@longaccess}%
6143 \expandafter
6144 \protected@xdef\csname glo@\@glo@label @longpluralaccess\endcsname{%
6145 \@glo@longpluralaccess}%
6146 }

```

## 5.2 Accessing Replacement Text

`\glsentryaccess` Get the value of the access key for the entry with the given label:

```

6147 \newcommand*\glsentryaccess}[1]{%
6148 \csname glo@#1@access\endcsname
6149 }

```

`\glsentrytextaccess` Get the value of the textaccess key for the entry with the given label:

```

6150 \newcommand*\glsentrytextaccess}[1]{%
6151 \csname glo@#1@textaccess\endcsname
6152 }

```

`\glsentryfirstaccess` Get the value of the firstaccess key for the entry with the given label:

```

6153 \newcommand*\glsentryfirstaccess}[1]{%
6154 \csname glo@#1@firstaccess\endcsname
6155 }

```

`\glsentrypluralaccess` Get the value of the pluralaccess key for the entry with the given label:

```

6156 \newcommand*\glsentrypluralaccess}[1]{%
6157 \csname glo@#1@pluralaccess\endcsname
6158 }

```

`\glsentryfirstpluralaccess` Get the value of the firstpluralaccess key for the entry with the given label:

```

6159 \newcommand*\glsentryfirstpluralaccess}[1]{%
6160 \csname glo@#1@firstpluralaccess\endcsname
6161 }

```

`\glsentrysymbolaccess` Get the value of the symbolaccess key for the entry with the given label:

```

6162 \newcommand*\glsentrysymbolaccess}[1]{%
6163 \csname glo@#1@symbolaccess\endcsname
6164 }

```

`\glsentrysymbolpluralaccess` Get the value of the symbolpluralaccess key for the entry with the given label:

```

6165 \newcommand*\glsentrysymbolpluralaccess}[1]{%
6166 \csname glo@#1@symbolpluralaccess\endcsname
6167 }

```

`\glsentrydescaccess` Get the value of the descriptionaccess key for the entry with the given label:

```

6168 \newcommand*\glsentrydescaccess}[1]{%
6169 \csname glo@#1@descaccess\endcsname
6170 }

```

trydescpluralaccess    Get the value of the descriptionpluralaccess key for the entry with the given label:

```
6171 \newcommand*{\glentrydescpluralaccess}[1]{%
6172   \csname glo@#1@descaccess\endcsname
6173 }
```

glentryshortaccess    Get the value of the shortaccess key for the entry with the given label:

```
6174 \newcommand*{\glentryshortaccess}[1]{%
6175   \csname glo@#1@shortaccess\endcsname
6176 }
```

ryshortpluralaccess    Get the value of the shortpluralaccess key for the entry with the given label:

```
6177 \newcommand*{\glentryshortpluralaccess}[1]{%
6178   \csname glo@#1@shortpluralaccess\endcsname
6179 }
```

\glentrylongaccess    Get the value of the longaccess key for the entry with the given label:

```
6180 \newcommand*{\glentrylongaccess}[1]{%
6181   \csname glo@#1@longaccess\endcsname
6182 }
```

trylongpluralaccess    Get the value of the longpluralaccess key for the entry with the given label:

```
6183 \newcommand*{\glentrylongpluralaccess}[1]{%
6184   \csname glo@#1@longpluralaccess\endcsname
6185 }
```

\glsacccsupp    \glsacccsupp{<replacement text>}{<text>}

This can be redefined to use E or Alt instead of ActualText. (I don't have the software to test the E or Alt options.)

```
6186 \newcommand*{\glsacccsupp}[2]{%
6187   \BeginAccSupp{ActualText=#1}#2\EndAccSupp}%
6188 }
```

\xglsacccsupp    Fully expands replacement text before calling \glsacccsupp

```
6189 \newcommand*{\xglsacccsupp}[2]{%
6190   \protected@edef\@gls@replacementtext{#1}%
6191   \expandafter\glsacccsupp\expandafter{\@gls@replacementtext}{#2}%
6192 }
```

lnameaccessdisplay    Displays the first argument with the accessibility text for the entry with the label given by the second argument (if set).

```
6193 \DeclareRobustCommand*{\glnameaccessdisplay}[2]{%
6194   \protected@edef\@glo@access{\glentryaccess{#2}}%
6195   \ifx\@glo@access\@gls@noaccess
6196     #1%
6197   \else
6198     \xglsacccsupp{\@glo@access}{#1}%
6199   \fi}
```

```

6199 \fi
6200 }

```

`lsttextaccessdisplay` As above but for the `textaccess` replacement text.

```

6201 \DeclareRobustCommand*\glstextaccessdisplay}[2]{%
6202   \protected@edef\@glo@access{\glstentrytextaccess{#2}}%
6203   \ifx\@glo@access\@gls@noaccess
6204     #1%
6205   \else
6206     \xglsaccsupp{\@glo@access}{#1}%
6207   \fi
6208 }

```

`pluralaccessdisplay` As above but for the `pluralaccess` replacement text.

```

6209 \DeclareRobustCommand*\glspluralaccessdisplay}[2]{%
6210   \protected@edef\@glo@access{\glsentrypluralaccess{#2}}%
6211   \ifx\@glo@access\@gls@noaccess
6212     #1%
6213   \else
6214     \xglsaccsupp{\@glo@access}{#1}%
6215   \fi
6216 }

```

`firstaccessdisplay` As above but for the `firstaccess` replacement text.

```

6217 \DeclareRobustCommand*\glsfirstaccessdisplay}[2]{%
6218   \protected@edef\@glo@access{\glsentryfirstaccess{#2}}%
6219   \ifx\@glo@access\@gls@noaccess
6220     #1%
6221   \else
6222     \xglsaccsupp{\@glo@access}{#1}%
6223   \fi
6224 }

```

`pluralaccessdisplay` As above but for the `firstpluralaccess` replacement text.

```

6225 \DeclareRobustCommand*\glsfirstpluralaccessdisplay}[2]{%
6226   \protected@edef\@glo@access{\glsentryfirstpluralaccess{#2}}%
6227   \ifx\@glo@access\@gls@noaccess
6228     #1%
6229   \else
6230     \xglsaccsupp{\@glo@access}{#1}%
6231   \fi
6232 }

```

`symbolaccessdisplay` As above but for the `symbolaccess` replacement text.

```

6233 \DeclareRobustCommand*\glssymbolaccessdisplay}[2]{%
6234   \protected@edef\@glo@access{\glsentrysymbolaccess{#2}}%
6235   \ifx\@glo@access\@gls@noaccess
6236     #1%
6237   \else

```

```

6238 \xglsaccsupp{\@glo@access}{#1}%
6239 \fi
6240 }

```

pluralaccessdisplay As above but for the symbolpluralaccess replacement text.

```

6241 \DeclareRobustCommand*\glssymbolpluralaccessdisplay}[2]{%
6242 \protected@edef\@glo@access{\glentrysymbolpluralaccess{#2}}%
6243 \ifx\@glo@access\@gls@noaccess
6244 #1%
6245 \else
6246 \xglsaccsupp{\@glo@access}{#1}%
6247 \fi
6248 }

```

descriptionaccessdisplay As above but for the descriptionaccess replacement text.

```

6249 \DeclareRobustCommand*\glsdescriptionaccessdisplay}[2]{%
6250 \protected@edef\@glo@access{\glentrydescaccess{#2}}%
6251 \ifx\@glo@access\@gls@noaccess
6252 #1%
6253 \else
6254 \xglsaccsupp{\@glo@access}{#1}%
6255 \fi
6256 }

```

descriptionpluralaccessdisplay As above but for the descriptionpluralaccess replacement text.

```

6257 \DeclareRobustCommand*\glsdescriptionpluralaccessdisplay}[2]{%
6258 \protected@edef\@glo@access{\glentrydescpluralaccess{#2}}%
6259 \ifx\@glo@access\@gls@noaccess
6260 #1%
6261 \else
6262 \xglsaccsupp{\@glo@access}{#1}%
6263 \fi
6264 }

```

shortaccessdisplay As above but for the shortaccess replacement text.

```

6265 \DeclareRobustCommand*\glsshortaccessdisplay}[2]{%
6266 \protected@edef\@glo@access{\glentryshortaccess{#2}}%
6267 \ifx\@glo@access\@gls@noaccess
6268 #1%
6269 \else
6270 \xglsaccsupp{\@glo@access}{#1}%
6271 \fi
6272 }

```

shortpluralaccessdisplay As above but for the shortpluralaccess replacement text.

```

6273 \DeclareRobustCommand*\glsshortpluralaccessdisplay}[2]{%
6274 \protected@edef\@glo@access{\glentryshortpluralaccess{#2}}%
6275 \ifx\@glo@access\@gls@noaccess
6276 #1%

```

```

6277 \else
6278 \xglsaccsupp{\@glo@access}{#1}%
6279 \fi
6280 }

```

`\glslongaccessdisplay` As above but for the longaccess replacement text.

```

6281 \DeclareRobustCommand*\glslongaccessdisplay}[2]{%
6282 \protected@edef\@glo@access{\glsentrylongaccess{#2}}%
6283 \ifx\@glo@access\@gls@noaccess
6284 #1%
6285 \else
6286 \xglsaccsupp{\@glo@access}{#1}%
6287 \fi
6288 }

```

`\glspluralaccessdisplay` As above but for the longpluralaccess replacement text.

```

6289 \DeclareRobustCommand*\glslongpluralaccessdisplay}[2]{%
6290 \protected@edef\@glo@access{\glsentrylongpluralaccess{#2}}%
6291 \ifx\@glo@access\@gls@noaccess
6292 #1%
6293 \else
6294 \xglsaccsupp{\@glo@access}{#1}%
6295 \fi
6296 }

```

`\glsaccessdisplay` Gets the replacement text corresponding to the named key given by the first argument and calls the appropriate command defined above.

```

6297 \DeclareRobustCommand*\glsaccessdisplay}[3]{%
6298 \@ifundefined{gls#1accessdisplay}%
6299 {%
6300 \PackageError{glossaries-accsupp}{No accessibility support
6301 for key ‘#1’}{}%
6302 }%
6303 {%
6304 \csname gls#1accessdisplay\endcsname{#2}{#3}%
6305 }%
6306 }

```

`\@gls@` Redefine `\@gls@` to change the way the link text is defined

```

6307 \def\@gls@#1#2[#3]{%
6308 \glsdoifexists{#2}%
6309 {%
6310 \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

6311 \def\@gls@link@opts{#1}%
6312 \def\@gls@link@label{#2}%

```

Determine what the link text should be (this is stored in \@glo@text). This is no longer expanded.

```

6313 \ifglsused{#2}%
6314 {%
6315 \def\@glo@text{\csname gls@\@glo@type @display\endcsname
6316 {\glsfirstaccessdisplay{\glsentrytext{#2}}{#2}}%
6317 {\glsdescriptionaccessdisplay{\glsentrydesc{#2}}{#2}}%
6318 {\glsymbolaccessdisplay{\glsentrysymbol{#2}}{#2}}%
6319 {#3}}%
6320 }%
6321 {%
6322 \def\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
6323 {\glsfirstaccessdisplay{\glsentryfirst{#2}}{#2}}%
6324 {\glsdescriptionaccessdisplay{\glsentrydesc{#2}}{#2}}%
6325 {\glsymbolaccessdisplay{\glsentrysymbol{#2}}{#2}}%
6326 {#3}}%
6327 }%

```

Call \@gls@link. If footnote package option has been used, suppress hyperlink for first use.

```

6328 \ifglsused{#2}%
6329 {%
6330 \@gls@link[#1]{#2}{\@glo@text}%
6331 }%
6332 {%
6333 \gls@checkisacronymlist\@glo@type
6334 \ifthenelse{(\boolean{@glsisacronymlist})\AND
6335 \boolean{glsacrfootnote}) \OR\NOT\boolean{glshyperfirst}}{%
6336 {%
6337 \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
6338 }%
6339 {%
6340 \@gls@link[#1]{#2}{\@glo@text}%
6341 }%
6342 }%

```

Indicate that this entry has now been used

```

6343 \glsunset{#2}%
6344 }%
6345 }

```

\@Gls@

```

6346 \def\@Gls@#1#2[#3]{%
6347 \glsdoifexists{#2}%
6348 {%
6349 \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

6350 \def\@gls@link@opts{#1}%
6351 \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

```

6352   \ifglsused{#2}%
6353   {%
6354     \def\@glo@text{\csname gls@\@glo@type @display\endcsname
6355       {\glsfirstaccessdisplay{\Glsentrytext{#2}}{#2}}}%
6356     {\glsdescriptionaccessdisplay{\glsentrydesc{#2}}{#2}}}%
6357     {\glsymbolaccessdisplay{\glsentrysymbol{#2}}{#2}}}%
6358     {#3}}}%
6359   }%
6360   {%
6361     \def\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
6362       {\glsfirstaccessdisplay{\Glsentryfirst{#2}}{#2}}}%
6363     {\glsdescriptionaccessdisplay{\glsentrydesc{#2}}{#2}}}%
6364     {\glsymbolaccessdisplay{\glsentrysymbol{#2}}{#2}}}%
6365     {#3}}}%
6366   }%

```

Call \@gls@link. If footnote package option has been used, suppress hyperlink for first use.

```

6367   \ifglsused{#2}%
6368   {%
6369     \@gls@link[#1]{#2}{\@glo@text}%
6370   }%
6371   {%
6372     \gls@checkisacronymlist\@glo@type
6373     \ifthenelse{(\boolean{glsisacronymlist})\AND
6374       \boolean{glsacrfootnote}) \OR\NOT\boolean{glshyperfirst}}{%
6375       {%
6376         \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
6377       }%
6378       {%
6379         \@gls@link[#1]{#2}{\@glo@text}%
6380       }%
6381     }%

```

Indicate that this entry has now been used

```

6382   \glsunset{#2}%
6383   }%
6384 }

```

\@GLS@

```

6385 \def\@GLS@#1#2[#3]{%
6386   \glsdoifexists{#2}{%
6387     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

6388   \def\@gls@link@opts{#1}%
6389   \def\@gls@link@label{#2}%

```

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

```

6390 \ifglsused{#2}%
6391 {%
6392 \def\@glo@text{\csname gls@\@glo@type @display\endcsname
6393 {\glstextaccessdisplay{\glentrytext{#2}}{#2}}%
6394 {\glsdescriptionaccessdisplay{\glentrydesc{#2}}{#2}}%
6395 {\glssymbolaccessdisplay{\glentrysymbol{#2}}{#2}}%
6396 {#3}}%
6397 }%
6398 {%
6399 \edef\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
6400 {\glsfirstaccessdisplay{\glentryfirst{#2}}{#2}}%
6401 {\glsdescriptionaccessdisplay{\glentrydesc{#2}}{#2}}%
6402 {\glssymbolaccessdisplay{\glentrysymbol{#2}}{#2}}%
6403 {#3}}%
6404 }%

```

Call \@gls@link If footnote package option has been used, suppress hyperlink for first use.

```

6405 \ifglsused{#2}%
6406 {%
6407 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
6408 }%
6409 {%
6410 \gls@checkisacronymlist\@glo@type
6411 \ifthenelse{\boolean{glsisacronymlist}\AND
6412 \boolean{glsacrfootnote}}{\OR\NOT\boolean{glshyperfirst}}{%
6413 \@gls@link[#1,hyper=false]{#2}{\MakeUppercase{\@glo@text}}%
6414 }%
6415 {%
6416 \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
6417 }%
6418 }%

```

Indicate that this entry has now been used

```

6419 \glsunset{#2}%
6420 }%
6421 }

```

\@gls@pl@

```

6422 \def\@glspl@#1#2[#3]{%
6423 \glsdoifexists{#2}%
6424 {%
6425 \edef\@glo@type{\glentrytype{#2}}%

```

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

```

6426 \def\@gls@link@opts{#1}%
6427 \def\@gls@link@label{#2}%

```

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

```

6428 \ifglsused{#2}%
6429 {%
6430   \def\@glo@text{\csname gls@\@glo@type @display\endcsname
6431     {\glspluralaccessdisplay{\glsentryplural{#2}}{#2}}%
6432     {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
6433     {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
6434     {#3}}%
6435   }%
6436   {%
6437     \def\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
6438       {\glsfirstpluralaccessdisplay{\glsentryfirstplural{#2}}{#2}}%
6439       {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
6440       {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
6441       {#3}}%
6442     }%

```

Call \@gls@link If footnote package option has been used, suppress hyperlink for first use.

```

6443 \ifglsused{#2}%
6444 {%
6445   \@gls@link[#1]{#2}{\@glo@text}%
6446 }%
6447 {%
6448   \gls@checkisacronymlist\@glo@type
6449   \ifthenelse{\boolean{glsisacronymlist}\AND
6450     \boolean{glsacrfootnote}}{\OR\NOT\boolean{glshyperfirst}}%
6451   {%
6452     \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
6453   }%
6454   {%
6455     \@gls@link[#1]{#2}{\@glo@text}%
6456   }%
6457 }%

```

Indicate that this entry has now been used

```

6458 \glsunset{#2}%
6459 }%
6460 }

```

\@Glspl@

```

6461 \def\@Glspl@#1#2[#3]{%
6462   \glsdoifexists{#2}%
6463   {%
6464     \edef\@glo@type{\glsentrytype{#2}}%
6465     \def\@gls@link@opts{#1}%
6466     \def\@gls@link@label{#2}%

```

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

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

```

6467 \ifglsused{#2}%

```

```

6468   {%
6469     \def\@glo@text{\csname gls@\@glo@type @display\endcsname
6470       {\glspluralaccessdisplay{\Glsentryplural{#2}}{#2}}%
6471       {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
6472       {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
6473       {#3}}%
6474   }%
6475   {%
6476     \def\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
6477       {\glsfirstpluralaccessdisplay{\Glsentryfirstplural{#2}}{#2}}%
6478       {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
6479       {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
6480       {#3}}%
6481   }%

```

Call \@gls@link If footnote package option has been used, suppress hyperlink for first use.

```

6482   \ifglsused{#2}%
6483   {%
6484     \@gls@link[#1]{#2}{\@glo@text}%
6485   }%
6486   {%
6487     \ifthenelse{\equal{\@glo@type}{\acronymtype}\and
6488       \boolean{glsacrfootnote}}%
6489     {%
6490       \@gls@link[#1,hyper=false]{#2}{\@glo@text}%
6491     }%
6492     {%
6493       \@gls@link[#1]{#2}{\@glo@text}%
6494     }%
6495   }%

```

Indicate that this entry has now been used

```

6496   \glsunset{#2}%
6497 }%
6498 }

```

\@GLSpl@

```

6499 \def\@GLSpl@#1#2[#3]{%
6500   \glsdoifexists{#2}%
6501   {%
6502     \edef\@glo@type{\glsentrytype{#2}}%

```

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

```

6503   \def\@gls@link@opts{#1}%
6504   \def\@gls@link@label{#2}%

```

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

```

6505   \ifglsused{#2}%
6506   {%
6507     \def\@glo@text{\csname gls@\@glo@type @display\endcsname

```

```

6508      {\glspluralaccessdisplay{\glsentryplural{#2}}{#2}}%
6509      {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
6510      {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
6511      {#3}}%
6512  }%
6513  {%
6514      \def\@glo@text{\csname gls@\@glo@type @displayfirst\endcsname
6515      {\glsfirstpluralaccessdisplay{\glsentryfirstplural{#2}}{#2}}%
6516      {\glsdescriptionpluralaccessdisplay{\glsentrydescplural{#2}}{#2}}%
6517      {\glsymbolpluralaccessdisplay{\glsentrysymbolplural{#2}}{#2}}%
6518      {#3}}%
6519  }%

```

Call \@gls@link If footnote package option has been used, suppress hyperlink for first use.

```

6520      \ifglsused{#2}%
6521      {%
6522          \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
6523      }%
6524      {%
6525          \gls@checkisacronymlist\@glo@type
6526          \ifthenelse{\(\boolean{@glsisacronymlist}\AND
6527              \boolean{glsacrfootnote})\OR\NOT\boolean{glshyperfirst}}%
6528              {%
6529                  \@gls@link[#1,hyper=false]{#2}{\MakeUppercase{\@glo@text}}%
6530              }%
6531              {%
6532                  \@gls@link[#1]{#2}{\MakeUppercase{\@glo@text}}%
6533              }%
6534      }%

```

Indicate that this entry has now been used

```

6535      \glsunset{#2}%
6536  }%
6537 }

```

\@acrshort

```

6538 \def\@acrshort#1#2[#3]{%
6539     \glsdoifexists{#2}%
6540     {%
6541         \edef\@glo@type{\glsentrytype{#2}}%
6542         \def\@glo@text{%
6543             \glsshortaccessdisplay{\glsentryshort{#2}}{#2}%
6544         }%
6545         Determine what the link text should be (this is stored in \@glo@text)
6546         Call \@gls@link
6547         \@gls@link[#1]{#2}{\acronymfont{\@glo@text}#3}%
6548     }%
6549 }

```

\@Acrshort

```
6548 \def\@Acrshort#1#2[#3]{%
6549   \glsdoifexists{#2}%
6550   {%
6551     \edef\@glo@type{\glsentrytype{#2}}%
        Determine what the link text should be (this is stored in \@glo@text)
6552     \def\@glo@text{%
6553       \glsshortaccessdisplay{\Glsentryshort{#2}}{#2}%
6554     }%
        Call \@gls@link
6555     \@gls@link[#1]{#2}{\acronymfont{\@glo@text}#3}%
6556   }%
6557 }
```

\@ACRshort

```
6558 \def\@ACRshort#1#2[#3]{%
6559   \glsdoifexists{#2}%
6560   {%
6561     \edef\@glo@type{\glsentrytype{#2}}%
        Determine what the link text should be (this is stored in \@glo@text)
6562     \def\@glo@text{%
6563       \glsshortaccessdisplay{\MakeUppercase{\glsentryshort{#2}}}{#2}%
6564     }%
        Call \@gls@link
6565     \@gls@link[#1]{#2}{\acronymfont{\@glo@text}#3}%
6566   }%
6567 }
```

\@acrlong

```
6568 \def\@acrlong#1#2[#3]{%
6569   \glsdoifexists{#2}%
6570   {%
6571     \edef\@glo@type{\glsentrytype{#2}}%
        Determine what the link text should be (this is stored in \@glo@text)
6572     \def\@glo@text{%
6573       \glslongaccessdisplay{\glsentrylong{#2}}{#2}%
6574     }%
        Call \@gls@link
6575     \@gls@link[#1]{#2}{\@glo@text}#3}%
6576   }%
6577 }
```

\@Acrlong

```
6578 \def\@Acrlong#1#2[#3]{%
6579   \glsdoifexists{#2}%
```

```

6580  {%
6581    \edef\@glo@type{\glentrytype{#2}}%
        Determine what the link text should be (this is stored in \@glo@text)
6582    \def\@glo@text{%
6583      \glslongaccessdisplay{\Glsentrylong{#2}}{#2}%
6584    }%
        Call \@gls@link
6585    \@gls@link[#1]{#2}{\@glo@text#3}%
6586  }%
6587 }

```

\@ACRlong

```

6588 \def\@ACRlong#1#2[#3]{%
6589   \glsdoifexists{#2}%
6590   {%
6591     \edef\@glo@type{\glentrytype{#2}}%
        Determine what the link text should be (this is stored in \@glo@text)
6592     \def\@glo@text{%
6593       \glslongaccessdisplay{\MakeUppercase{\glentrylong{#2}}}{#2}%
6594     }%
        Call \@gls@link
6595     \@gls@link[#1]{#2}{\@glo@text#3}%
6596   }%
6597 }

```

### 5.3 Displaying the Glossary

Entries within the glossary or list of acronyms are now formatted via `\accsuppglossaryentryfield` and `\accsuppglossarysubentryfield`.

`@glossaryentryfield`

```

6598 \ifglxindy
6599   \renewcommand*{\@glossaryentryfield}{%
6600     \string\accsuppglossaryentryfield}
6601 \else
6602   \renewcommand*{\@glossaryentryfield}{%
6603     \string\accsuppglossaryentryfield}
6604 \fi

```

`ossarysubentryfield`

```

6605 \ifglxindy
6606   \renewcommand*{\@glossarysubentryfield}{%
6607     \string\accsuppglossarysubentryfield}
6608 \else
6609   \renewcommand*{\@glossarysubentryfield}{%
6610     \string\accsuppglossarysubentryfield}
6611 \fi

```

pglossaryentryfield

```

6612 \newcommand*{\accsuppglossaryentryfield}[5]{%
6613   \glossaryentryfield{#1}%
6614   {\glsnameaccessdisplay{#2}{#1}}%
6615   {\glsdescriptionaccessdisplay{#3}{#1}}%
6616   {\glsymbolaccessdisplay{#4}{#1}}{#5}%
6617 }

```

ossarysubentryfield

```

6618 \newcommand*{\accsuppglossarysubentryfield}[6]{%
6619   \glossaryentryfield{#1}{#2}%
6620   {\glsnameaccessdisplay{#3}{#2}}%
6621   {\glsdescriptionaccessdisplay{#4}{#2}}%
6622   {\glsymbolaccessdisplay{#5}{#2}}{#6}%
6623 }

```

## 5.4 Acronyms

Use `\newacronymhook` to modify the key list to set the access text to the long version by default.

```

6624 \renewcommand*{\newacronymhook}{%
6625   \edef\@gls@keylist{shortaccess=\the\glslongtok,%
6626     \the\glskeylisttok}%
6627   \expandafter\glskeylisttok\expandafter{\@gls@keylist}%
6628 }

```

efaultNewAcronymDef Modify default style to use access text:

```

6629 \renewcommand*{\DefaultNewAcronymDef}{%
6630   \edef\@do@newglossaryentry{%
6631     \noexpand\newglossaryentry{\the\glslabeltok}%
6632     {%
6633       type=\acronymtype,%
6634       name={\the\glsshorttok},%
6635       description={\the\glslongtok},%
6636       descriptionaccess=\relax,%
6637       text={\the\glsshorttok},%
6638       access={\noexpand\@glo@textaccess},%
6639       sort={\the\glsshorttok},%
6640       short={\the\glsshorttok},%
6641       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6642       shortaccess={\the\glslongtok},%
6643       long={\the\glslongtok},%
6644       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6645       descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6646       first={\noexpand\glslongaccessdisplay
6647         {\the\glslongtok}{\the\glslabeltok}\space
6648         (\noexpand\glsshortaccessdisplay
6649           {\the\glsshorttok}{\the\glslabeltok})},%

```

```

6650 plural={\the\glsshorttok\acrpluralsuffix},%
6651 firstplural={\noexpand\glslongpluralaccessdisplay
6652   {\noexpand\@glo@longpl}{\the\glslabeltok}\space
6653   (\noexpand\glsshortpluralaccessdisplay
6654     {\noexpand\@glo@shortpl}{\the\glslabeltok})},%
6655 firstaccess=\relax,
6656 firstpluralaccess=\relax,
6657 textaccess={\noexpand\@glo@shortaccess},%
6658 \the\glskeylisttok
6659 }%
6660 }%
6661 \@do@newglossaryentry
6662 }

```

otnoteNewAcronymDef

```

6663 \renewcommand*{\DescriptionFootnoteNewAcronymDef}{%
6664   \edef\@do@newglossaryentry{%
6665     \noexpand\newglossaryentry{\the\glslabeltok}%
6666     {%
6667       type=\acronymtype,%
6668       name={\noexpand\acronymfont{\the\glsshorttok}},%
6669       sort={\the\glsshorttok},%
6670       text={\the\glsshorttok},%
6671       short={\the\glsshorttok},%
6672       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6673       shortaccess={\the\glslongtok},%
6674       long={\the\glslongtok},%
6675       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6676       access={\noexpand\@glo@textaccess},%
6677       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6678       symbol={\the\glslongtok},%
6679       symbolplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6680       firstpluralaccess=\relax,
6681       textaccess={\noexpand\@glo@shortaccess},%
6682       \the\glskeylisttok
6683     }%
6684   }%
6685   \@do@newglossaryentry
6686 }

```

iptionNewAcronymDef

```

6687 \renewcommand*{\DescriptionNewAcronymDef}{%
6688   \edef\@do@newglossaryentry{%
6689     \noexpand\newglossaryentry{\the\glslabeltok}%
6690     {%
6691       type=\acronymtype,%
6692       name={\noexpand
6693         \acronymformat{\the\glsshorttok}{\the\glslongtok}},%
6694       access={\noexpand\@glo@textaccess},%

```

```

6695     sort={\the\glsshorttok},%
6696     short={\the\glsshorttok},%
6697     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6698     shortaccess={\the\glslongtok},%
6699     long={\the\glslongtok},%
6700     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6701     first={\the\glslongtok},%
6702     firstaccess=\relax,
6703     firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6704     text={\the\glsshorttok},%
6705     textaccess={\the\glslongtok},%
6706     plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6707     symbol={\noexpand\@glo@text},%
6708     symbolaccess={\noexpand\@glo@textaccess},%
6709     symbolplural={\noexpand\@glo@plural},%
6710     firstpluralaccess=\relax,
6711     textaccess={\noexpand\@glo@shortaccess},%
6712     \the\glskeylisttok}%
6713 }%
6714 \@do@newglossaryentry
6715 }

```

otnoteNewAcronymDef

```

6716 \renewcommand*{\FootnoteNewAcronymDef}{%
6717   \edef\@do@newglossaryentry{%
6718     \noexpand\newglossaryentry{\the\glslabeltok}%
6719     {%
6720       type=\acronymtype,%
6721       name={\noexpand\acronymfont{\the\glsshorttok}},%
6722       sort={\the\glsshorttok},%
6723       text={\the\glsshorttok},%
6724       textaccess={\the\glslongtok},%
6725       access={\noexpand\@glo@textaccess},%
6726       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6727       short={\the\glsshorttok},%
6728       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6729       long={\the\glslongtok},%
6730       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6731       description={\the\glslongtok},%
6732       descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6733       \the\glskeylisttok
6734     }%
6735   }%
6736   \@do@newglossaryentry
6737 }

```

\SmallNewAcronymDef

```

6738 \renewcommand*{\SmallNewAcronymDef}{%
6739   \edef\@do@newglossaryentry{%

```

```

6740 \noexpand\newglossaryentry{\the\glslabeltok}%
6741 {%
6742   type=\acronymtype,%
6743   name={\noexpand\acronymfont{\the\glsshorttok}},%
6744   access={\noexpand\@glo@symbolaccess},%
6745   sort={\the\glsshorttok},%
6746   short={\the\glsshorttok},%
6747   shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6748   shortaccess={\the\glslongtok},%
6749   long={\the\glslongtok},%
6750   longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6751   text={\noexpand\@glo@short},%
6752   textaccess={\noexpand\@glo@shortaccess},%
6753   plural={\noexpand\@glo@shortpl},%
6754   first={\the\glslongtok},%
6755   firstaccess=\relax,%
6756   firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6757   description={\noexpand\@glo@first},%
6758   descriptionplural={\noexpand\@glo@firstplural},%
6759   symbol={\the\glsshorttok},%
6760   symbolaccess={\the\glslongtok},%
6761   symbolplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6762   \the\glskeylisttok
6763 }%
6764 }%
6765 \@do@newglossaryentry
6766 }

```

The following are kept for compatibility with versions before 3.0:

```

\glsshortaccesskey
6767 \newcommand*{\glsshortaccesskey}{\glsshortkey access}%

hortpluralaccesskey
6768 \newcommand*{\glsshortpluralaccesskey}{\glsshortpluralkey access}%

\glslongaccesskey
6769 \newcommand*{\glslongaccesskey}{\glslongkey access}%

longpluralaccesskey
6770 \newcommand*{\glslongpluralaccesskey}{\glslongpluralkey access}%

```

## 5.5 Debugging Commands

```

\showglongnameaccess
6771 \newcommand*{\showglongnameaccess}[1]{%
6772   \expandafter\show\csname glo@#1@textaccess\endcsname
6773 }

```

```

\showglotextaccess
6774 \newcommand*{\showglotextaccess}[1]{%
6775   \expandafter\show\csname glo@#1@textaccess\endcsname
6776 }

showglopluralaccess
6777 \newcommand*{\showglopluralaccess}[1]{%
6778   \expandafter\show\csname glo@#1@pluralaccess\endcsname
6779 }

\showglofirstaccess
6780 \newcommand*{\showglofirstaccess}[1]{%
6781   \expandafter\show\csname glo@#1@firstaccess\endcsname
6782 }

lofirstpluralaccess
6783 \newcommand*{\showglofirstpluralaccess}[1]{%
6784   \expandafter\show\csname glo@#1@firstpluralaccess\endcsname
6785 }

showglosymbolaccess
6786 \newcommand*{\showglosymbolaccess}[1]{%
6787   \expandafter\show\csname glo@#1@symbolaccess\endcsname
6788 }

osymbolpluralaccess
6789 \newcommand*{\showglosymbolpluralaccess}[1]{%
6790   \expandafter\show\csname glo@#1@symbolpluralaccess\endcsname
6791 }

\showglodescaccess
6792 \newcommand*{\showglodescaccess}[1]{%
6793   \expandafter\show\csname glo@#1@descaccess\endcsname
6794 }

glodescpluralaccess
6795 \newcommand*{\showglodescpluralaccess}[1]{%
6796   \expandafter\show\csname glo@#1@descpluralaccess\endcsname
6797 }

\showgloshortaccess
6798 \newcommand*{\showgloshortaccess}[1]{%
6799   \expandafter\show\csname glo@#1@shortaccess\endcsname
6800 }

loshortpluralaccess
6801 \newcommand*{\showgloshortpluralaccess}[1]{%
6802   \expandafter\show\csname glo@#1@shortpluralaccess\endcsname
6803 }

```

```

\showglolongaccess
6804 \newcommand*\showglolongaccess}[1]{%
6805   \expandafter\show\csname glo@#1@longaccess\endcsname
6806 }

glolongpluralaccess
6807 \newcommand*\showglolongpluralaccess}[1]{%
6808   \expandafter\show\csname glo@#1@longpluralaccess\endcsname
6809 }

```

## 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.

```

6810 \NeedsTeXFormat{LaTeX2e}
6811 \ProvidesPackage{glossaries-babel}[2009/04/16 v1.2 (NLCT)]

English:
6812 \@ifundefined{captionsenglish}{\}{%
6813   \addto\captionsenglish{%
6814     \renewcommand*\glossaryname{Glossary}%
6815     \renewcommand*\acronymname{Acronyms}%
6816     \renewcommand*\entryname{Notation}%
6817     \renewcommand*\descriptionname{Description}%
6818     \renewcommand*\symbolname{Symbol}%
6819     \renewcommand*\pagelistname{Page List}%
6820     \renewcommand*\glssymbolsgroupname{Symbols}%
6821     \renewcommand*\glsnumbersgroupname{Numbers}%
6822 }%
6823 }
6824 \@ifundefined{captionsamerican}{\}{%
6825   \addto\captionsamerican{%
6826     \renewcommand*\glossaryname{Glossary}%
6827     \renewcommand*\acronymname{Acronyms}%
6828     \renewcommand*\entryname{Notation}%
6829     \renewcommand*\descriptionname{Description}%
6830     \renewcommand*\symbolname{Symbol}%
6831     \renewcommand*\pagelistname{Page List}%
6832     \renewcommand*\glssymbolsgroupname{Symbols}%
6833     \renewcommand*\glsnumbersgroupname{Numbers}%
6834 }%
6835 }
6836 \@ifundefined{captionsaustralian}{\}{%

```

```

6837 \addto\captionsaustrian{%
6838 \renewcommand*{\glossaryname}{Glossary}%
6839 \renewcommand*{\acronymname}{Acronyms}%
6840 \renewcommand*{\entryname}{Notation}%
6841 \renewcommand*{\descriptionname}{Description}%
6842 \renewcommand*{\symbolname}{Symbol}%
6843 \renewcommand*{\pagelistname}{Page List}%
6844 \renewcommand*{\glssymbolsgroupname}{Symbols}%
6845 \renewcommand*{\glsnumbersgroupname}{Numbers}%
6846 }%
6847 }
6848 \@ifundefined{captionsbritish}{-}{%
6849 \addto\captionsbritish{%
6850 \renewcommand*{\glossaryname}{Glossary}%
6851 \renewcommand*{\acronymname}{Acronyms}%
6852 \renewcommand*{\entryname}{Notation}%
6853 \renewcommand*{\descriptionname}{Description}%
6854 \renewcommand*{\symbolname}{Symbol}%
6855 \renewcommand*{\pagelistname}{Page List}%
6856 \renewcommand*{\glssymbolsgroupname}{Symbols}%
6857 \renewcommand*{\glsnumbersgroupname}{Numbers}%
6858 }%
6859 \@ifundefined{captionscanadian}{-}{%
6860 \addto\captionscanadian{%
6861 \renewcommand*{\glossaryname}{Glossary}%
6862 \renewcommand*{\acronymname}{Acronyms}%
6863 \renewcommand*{\entryname}{Notation}%
6864 \renewcommand*{\descriptionname}{Description}%
6865 \renewcommand*{\symbolname}{Symbol}%
6866 \renewcommand*{\pagelistname}{Page List}%
6867 \renewcommand*{\glssymbolsgroupname}{Symbols}%
6868 \renewcommand*{\glsnumbersgroupname}{Numbers}%
6869 }%
6870 }
6871 \@ifundefined{captionsnewzealand}{-}{%
6872 \addto\captionnewzealand{%
6873 \renewcommand*{\glossaryname}{Glossary}%
6874 \renewcommand*{\acronymname}{Acronyms}%
6875 \renewcommand*{\entryname}{Notation}%
6876 \renewcommand*{\descriptionname}{Description}%
6877 \renewcommand*{\symbolname}{Symbol}%
6878 \renewcommand*{\pagelistname}{Page List}%
6879 \renewcommand*{\glssymbolsgroupname}{Symbols}%
6880 \renewcommand*{\glsnumbersgroupname}{Numbers}%
6881 }%
6882 }
6883 \@ifundefined{captionsUKenglish}{-}{%
6884 \addto\captionUKenglish{%
6885 \renewcommand*{\glossaryname}{Glossary}%

```

```

6886 \renewcommand*{\acronymname}{Acronyms}%
6887 \renewcommand*{\entryname}{Notation}%
6888 \renewcommand*{\descriptionname}{Description}%
6889 \renewcommand*{\symbolname}{Symbol}%
6890 \renewcommand*{\pagelistname}{Page List}%
6891 \renewcommand*{\glssymbolsgroupname}{Symbols}%
6892 \renewcommand*{\glsnumbersgroupname}{Numbers}%
6893 }%
6894 }
6895 \@ifundefined{captionsUSenglish}{-}{%
6896 \addto\captionsUSenglish{%
6897 \renewcommand*{\glossaryname}{Glossary}%
6898 \renewcommand*{\acronymname}{Acronyms}%
6899 \renewcommand*{\entryname}{Notation}%
6900 \renewcommand*{\descriptionname}{Description}%
6901 \renewcommand*{\symbolname}{Symbol}%
6902 \renewcommand*{\pagelistname}{Page List}%
6903 \renewcommand*{\glssymbolsgroupname}{Symbols}%
6904 \renewcommand*{\glsnumbersgroupname}{Numbers}%
6905 }%
6906 }

```

German (quite a few variations were suggested for German; I settled on the following):

```

6907 \@ifundefined{captionsgerman}{-}{%
6908 \addto\captionsgerman{%
6909 \renewcommand*{\glossaryname}{Glossar}%
6910 \renewcommand*{\acronymname}{Akronyme}%
6911 \renewcommand*{\entryname}{Bezeichnung}%
6912 \renewcommand*{\descriptionname}{Beschreibung}%
6913 \renewcommand*{\symbolname}{Symbol}%
6914 \renewcommand*{\pagelistname}{Seiten}%
6915 \renewcommand*{\glssymbolsgroupname}{Symbole}%
6916 \renewcommand*{\glsnumbersgroupname}{Zahlen}}
6917 }

```

ngerman is identical to German:

```

6918 \@ifundefined{captionsgerman}{-}{%
6919 \addto\captionsgerman{%
6920 \renewcommand*{\glossaryname}{Glossar}%
6921 \renewcommand*{\acronymname}{Akronyme}%
6922 \renewcommand*{\entryname}{Bezeichnung}%
6923 \renewcommand*{\descriptionname}{Beschreibung}%
6924 \renewcommand*{\symbolname}{Symbol}%
6925 \renewcommand*{\pagelistname}{Seiten}%
6926 \renewcommand*{\glssymbolsgroupname}{Symbole}%
6927 \renewcommand*{\glsnumbersgroupname}{Zahlen}}
6928 }

```

Italian:

```

6929 \@ifundefined{captionsitalian}{-}{%

```

```

6930 \addto\captionssitalian{%
6931   \renewcommand*{\glossaryname}{Glossario}%
6932   \renewcommand*{\acronymname}{Acronimi}%
6933   \renewcommand*{\entryname}{Nomenclatura}%
6934   \renewcommand*{\descriptionname}{Descrizione}%
6935   \renewcommand*{\symbolname}{Simbolo}%
6936   \renewcommand*{\pagelistname}{Elenco delle pagine}%
6937   \renewcommand*{\glssymbolsgroupname}{Simboli}%
6938   \renewcommand*{\glsnumbersgroupname}{Numeri}}
6939 }

```

#### Dutch:

```

6940 \@ifundefined{captionssdutch}{}{%
6941   \addto\captionssdutch{%
6942     \renewcommand*{\glossaryname}{Woordenlijst}%
6943     \renewcommand*{\acronymname}{Acroniemen}%
6944     \renewcommand*{\entryname}{Benaming}%
6945     \renewcommand*{\descriptionname}{Beschrijving}%
6946     \renewcommand*{\symbolname}{Symbool}%
6947     \renewcommand*{\pagelistname}{Pagina's}%
6948     \renewcommand*{\glssymbolsgroupname}{Symbolen}%
6949     \renewcommand*{\glsnumbersgroupname}{Cijfers}}
6950 }

```

#### Spanish:

```

6951 \@ifundefined{captionssspanish}{}{%
6952   \addto\captionssspanish{%
6953     \renewcommand*{\glossaryname}{Glosario}%
6954     \renewcommand*{\acronymname}{Siglas}%
6955     \renewcommand*{\entryname}{Entrada}%
6956     \renewcommand*{\descriptionname}{Descripci\'on}%
6957     \renewcommand*{\symbolname}{S\'mbolo}%
6958     \renewcommand*{\pagelistname}{Lista de p\'aginas}%
6959     \renewcommand*{\glssymbolsgroupname}{S\'mbolos}%
6960     \renewcommand*{\glsnumbersgroupname}{N\'umeros}}
6961 }

```

#### French:

```

6962 \@ifundefined{captionssfrench}{}{%
6963   \addto\captionssfrench{%
6964     \renewcommand*{\glossaryname}{Glossaire}%
6965     \renewcommand*{\acronymname}{Acronymes}%
6966     \renewcommand*{\entryname}{Termes}%
6967     \renewcommand*{\descriptionname}{Description}%
6968     \renewcommand*{\symbolname}{Symbole}%
6969     \renewcommand*{\pagelistname}{Pages}%
6970     \renewcommand*{\glssymbolsgroupname}{Symboles}%
6971     \renewcommand*{\glsnumbersgroupname}{Nombres}}
6972 }
6973 \@ifundefined{captionssfrenchb}{}{%
6974   \addto\captionssfrenchb{%

```

```

6975 \renewcommand*{\glossaryname}{Glossaire}%
6976 \renewcommand*{\acronymname}{Acronymes}%
6977 \renewcommand*{\entryname}{Terme}%
6978 \renewcommand*{\descriptionname}{Description}%
6979 \renewcommand*{\symbolname}{Symbole}%
6980 \renewcommand*{\pagelistname}{Pages}%
6981 \renewcommand*{\glssymbolsgroupname}{Symboles}%
6982 \renewcommand*{\glsnumbersgroupname}{Nombres}}
6983 }
6984 \@ifundefined{captionsfrancais}{}{%
6985 \addto\captionsfrancais{%
6986 \renewcommand*{\glossaryname}{Glossaire}%
6987 \renewcommand*{\acronymname}{Acronymes}%
6988 \renewcommand*{\entryname}{Terme}%
6989 \renewcommand*{\descriptionname}{Description}%
6990 \renewcommand*{\symbolname}{Symbole}%
6991 \renewcommand*{\pagelistname}{Pages}%
6992 \renewcommand*{\glssymbolsgroupname}{Symboles}%
6993 \renewcommand*{\glsnumbersgroupname}{Nombres}}
6994 }

```

Danish:

```

6995 \@ifundefined{captionsdanish}{}{%
6996 \addto\captionsdanish{%
6997 \renewcommand*{\glossaryname}{Ordliste}%
6998 \renewcommand*{\acronymname}{Akronymer}%
6999 \renewcommand*{\entryname}{Symbolforklaring}%
7000 \renewcommand*{\descriptionname}{Beskrivelse}%
7001 \renewcommand*{\symbolname}{Symbol}%
7002 \renewcommand*{\pagelistname}{Side}%
7003 \renewcommand*{\glssymbolsgroupname}{Symboler}%
7004 \renewcommand*{\glsnumbersgroupname}{Tal}}
7005 }

```

Irish:

```

7006 \@ifundefined{captionsirish}{}{%
7007 \addto\captionsirish{%
7008 \renewcommand*{\glossaryname}{Gluais}%
7009 \renewcommand*{\acronymname}{Acrainmneacha}%

```

wasn't sure whether to go for Nóta (Note), Ciall ('Meaning', 'sense') or Brí ('Meaning'). In the end I chose Ciall.

```

7010 \renewcommand*{\entryname}{Ciall}%
7011 \renewcommand*{\descriptionname}{Tuairisc}%

```

Again, not sure whether to use Comhartha/Comharthaí or Siombail/Siombaile, so have chosen the former.

```

7012 \renewcommand*{\symbolname}{Comhartha}%
7013 \renewcommand*{\glssymbolsgroupname}{Comhartha\'}{\i}}%
7014 \renewcommand*{\pagelistname}{Leathanaigh}%
7015 \renewcommand*{\glsnumbersgroupname}{Uimhreacha}}

```

7016 }

#### Hungarian:

```
7017 \@ifundefined{captionsmagyar}{%}{%
7018   \addto\captionsmagyar{%
7019     \renewcommand*{\glossaryname}{Sz\`ojegyz\`ek}%
7020     \renewcommand*{\acronymname}{Bet\H uszavak}%
7021     \renewcommand*{\entryname}{Kifejez\`es}%
7022     \renewcommand*{\descriptionname}{Magyar\`azat}%
7023     \renewcommand*{\symbolname}{Jel\"ol\`es}%
7024     \renewcommand*{\pagelistname}{Oldalsz\`am}%
7025     \renewcommand*{\glssymbolsgroupname}{Jelek}%
7026     \renewcommand*{\glsnumbersgroupname}{Sz\`amjegyek}%
7027   }
7028 }
7029 \@ifundefined{captionshungarian}{%}{%
7030   \addto\captionshungarian{%
7031     \renewcommand*{\glossaryname}{Sz\`ojegyz\`ek}%
7032     \renewcommand*{\acronymname}{Bet\H uszavak}%
7033     \renewcommand*{\entryname}{Kifejez\`es}%
7034     \renewcommand*{\descriptionname}{Magyar\`azat}%
7035     \renewcommand*{\symbolname}{Jel\"ol\`es}%
7036     \renewcommand*{\pagelistname}{Oldalsz\`am}%
7037     \renewcommand*{\glssymbolsgroupname}{Jelek}%
7038     \renewcommand*{\glsnumbersgroupname}{Sz\`amjegyek}%
7039   }
7040 }
```

#### Polish

```
7041 \@ifundefined{captionspolish}{%}{%
7042   \addto\captionspolish{%
7043     \renewcommand*{\glossaryname}{S{\l}ownik termin\`ow}%
7044     \renewcommand*{\acronymname}{Skr\`ot}%
7045     \renewcommand*{\entryname}{Termin}%
7046     \renewcommand*{\descriptionname}{Opis}%
7047     \renewcommand*{\symbolname}{Symbol}%
7048     \renewcommand*{\pagelistname}{Strony}%
7049     \renewcommand*{\glssymbolsgroupname}{Symbole}%
7050     \renewcommand*{\glsnumbersgroupname}{Liczby}}
7051 }
```

#### Brazilian

```
7052 \@ifundefined{captionsbrazil}{%}{%
7053   \addto\captionsbrazil{%
7054     \renewcommand*{\glossaryname}{Gloss\`ario}%
7055     \renewcommand*{\acronymname}{Siglas}%
7056     \renewcommand*{\entryname}{Nota\c c\~ao}%
7057     \renewcommand*{\descriptionname}{Descri\c c\~ao}%
7058     \renewcommand*{\symbolname}{S\`imbolo}%
7059     \renewcommand*{\pagelistname}{Lista de P\`aginas}%
7060     \renewcommand*{\glssymbolsgroupname}{S\`imbolos}%

```

```

7061 \renewcommand*{\glsnumbersgroupname}{N\,umeros}%
7062 }%
7063 }

```

## 6.2 Polyglossia Captions

```

7064 \NeedsTeXFormat{LaTeX2e}
7065 \ProvidesPackage{glossaries-polyglossia}[2009/11/09 v1.0 (NLCT)]

```

English:

```

7066 \@ifundefined{captionsenglish}{}{%
7067   \expandafter\toks@\expandafter{\captionsenglish
7068     \renewcommand*{\glossaryname}{\textenglish{Glossary}}%
7069     \renewcommand*{\acronymname}{\textenglish{Acronyms}}%
7070     \renewcommand*{\entryname}{\textenglish{Notation}}%
7071     \renewcommand*{\descriptionname}{\textenglish{Description}}%
7072     \renewcommand*{\symbolname}{\textenglish{Symbol}}%
7073     \renewcommand*{\pagelistname}{\textenglish{Page List}}%
7074     \renewcommand*{\glssymbolsgroupname}{\textenglish{Symbols}}%
7075     \renewcommand*{\glsnumbersgroupname}{\textenglish{Numbers}}%
7076   }%
7077   \edef\captionsenglish{\the\toks@}%
7078 }

```

German:

```

7079 \@ifundefined{captionsgerman}{}{%
7080   \expandafter\toks@\expandafter{\captionsgerman
7081     \renewcommand*{\glossaryname}{\textgerman{Glossar}}%
7082     \renewcommand*{\acronymname}{\textgerman{Akronyme}}%
7083     \renewcommand*{\entryname}{\textgerman{Bezeichnung}}%
7084     \renewcommand*{\descriptionname}{\textgerman{Beschreibung}}%
7085     \renewcommand*{\symbolname}{\textgerman{Symbol}}%
7086     \renewcommand*{\pagelistname}{\textgerman{Seiten}}%
7087     \renewcommand*{\glssymbolsgroupname}{\textgerman{Symbole}}%
7088     \renewcommand*{\glsnumbersgroupname}{\textgerman{Zahlen}}%
7089   }%
7090   \edef\captionsgerman{\the\toks@}%
7091 }

```

Italian:

```

7092 \@ifundefined{captionsitalian}{}{%
7093   \expandafter\toks@\expandafter{\captionsitalian
7094     \renewcommand*{\glossaryname}{\textitalian{Glossario}}%
7095     \renewcommand*{\acronymname}{\textitalian{Acronimi}}%
7096     \renewcommand*{\entryname}{\textitalian{Nomenclatura}}%
7097     \renewcommand*{\descriptionname}{\textitalian{Descrizione}}%
7098     \renewcommand*{\symbolname}{\textitalian{Simbolo}}%
7099     \renewcommand*{\pagelistname}{\textitalian{Elenco delle pagine}}%
7100     \renewcommand*{\glssymbolsgroupname}{\textitalian{Simboli}}%
7101     \renewcommand*{\glsnumbersgroupname}{\textitalian{Numeri}}%
7102   }%

```

```

7103 \edef\captionsitalian{\the\toks@}%
7104 }

```

#### Dutch:

```

7105 \@ifundefined{captionsdutch}{}{%
7106   \expandafter\toks@\expandafter{\captionsdutch
7107     \renewcommand*{\glossaryname}{\textdutch{Woordenlijst}}%
7108     \renewcommand*{\acronymname}{\textdutch{Acroniemen}}%
7109     \renewcommand*{\entryname}{\textdutch{Benaming}}%
7110     \renewcommand*{\descriptionname}{\textdutch{Beschrijving}}%
7111     \renewcommand*{\symbolname}{\textdutch{Symbool}}%
7112     \renewcommand*{\pagelistname}{\textdutch{Pagina's}}%
7113     \renewcommand*{\glssymbolsgroupname}{\textdutch{Symbolen}}%
7114     \renewcommand*{\glsnumbersgroupname}{\textdutch{Cijfers}}%
7115   }%
7116   \edef\captionsdutch{\the\toks@}%
7117 }

```

#### Spanish:

```

7118 \@ifundefined{captionsspanish}{}{%
7119   \expandafter\toks@\expandafter{\captionsspanish
7120     \renewcommand*{\glossaryname}{\textspanish{Glosario}}%
7121     \renewcommand*{\acronymname}{\textspanish{Siglas}}%
7122     \renewcommand*{\entryname}{\textspanish{Entrada}}%
7123     \renewcommand*{\descriptionname}{\textspanish{Descripci'on}}%
7124     \renewcommand*{\symbolname}{\textspanish{S'\i mbolo}}%
7125     \renewcommand*{\pagelistname}{\textspanish{Lista de p'aginas}}%
7126     \renewcommand*{\glssymbolsgroupname}{\textspanish{S'\i mbolos}}%
7127     \renewcommand*{\glsnumbersgroupname}{\textspanish{N'umeros}}%
7128   }%
7129   \edef\captionsspanish{\the\toks@}%
7130 }

```

#### French:

```

7131 \@ifundefined{captionsfrench}{}{%
7132   \expandafter\toks@\expandafter{\captionsfrench
7133     \renewcommand*{\glossaryname}{\textfrench{Glossaire}}%
7134     \renewcommand*{\acronymname}{\textfrench{Acronymes}}%
7135     \renewcommand*{\entryname}{\textfrench{Terme}}%
7136     \renewcommand*{\descriptionname}{\textfrench{Description}}%
7137     \renewcommand*{\symbolname}{\textfrench{Symbole}}%
7138     \renewcommand*{\pagelistname}{\textfrench{Pages}}%
7139     \renewcommand*{\glssymbolsgroupname}{\textfrench{Symboles}}%
7140     \renewcommand*{\glsnumbersgroupname}{\textfrench{Nombres}}%
7141   }%
7142   \edef\captionsfrench{\the\toks@}%
7143 }

```

#### Danish:

```

7144 \@ifundefined{captionsdanish}{}{%
7145   \expandafter\toks@\expandafter{\captionsdanish
7146     \renewcommand*{\glossaryname}{\textdanish{Ordliste}}%

```

```

7147 \renewcommand*{\acronymname}{\textdanish{Akronymer}}}%
7148 \renewcommand*{\entryname}{\textdanish{Symbolforklaring}}}%
7149 \renewcommand*{\descriptionname}{\textdanish{Beskrivelse}}}%
7150 \renewcommand*{\symbolname}{\textdanish{Symbol}}}%
7151 \renewcommand*{\pagelistname}{\textdanish{Side}}}%
7152 \renewcommand*{\glssymbolsgroupname}{\textdanish{Symboler}}}%
7153 \renewcommand*{\glsnumbersgroupname}{\textdanish{Tal}}}%
7154 }%
7155 \edef\captionsdanish{\the\toks@}%
7156 }

Irish:
7157 \@ifundefined{captionsirish}{}{%
7158 \expandafter\toks@\expandafter{\captionsirish
7159 \renewcommand*{\glossaryname}{\textirish{Gluais}}}%
7160 \renewcommand*{\acronymname}{\textirish{Acrainmneacha}}}%
7161 \renewcommand*{\entryname}{\textirish{Ciall}}}%
7162 \renewcommand*{\descriptionname}{\textirish{Tuirisc}}}%
7163 \renewcommand*{\symbolname}{\textirish{Comhartha}}}%
7164 \renewcommand*{\glssymbolsgroupname}{\textirish{Comhartha\,'{i}}}%
7165 \renewcommand*{\pagelistname}{\textirish{Leathanaigh}}}%
7166 \renewcommand*{\glsnumbersgroupname}{\textirish{Uimhreacha}}}%
7167 }%
7168 \edef\captionsirish{\the\toks@}%
7169 }

Hungarian:
7170 \@ifundefined{captionsmagyar}{}{%
7171 \expandafter\toks@\expandafter{\captionsmagyar
7172 \renewcommand*{\glossaryname}{\textmagyar{Sz\,'ojegy\,'ek}}}%
7173 \renewcommand*{\acronymname}{\textmagyar{Bet\H uszavak}}}%
7174 \renewcommand*{\entryname}{\textmagyar{Kifejez\,'es}}}%
7175 \renewcommand*{\descriptionname}{\textmagyar{Magyar\,'azat}}}%
7176 \renewcommand*{\symbolname}{\textmagyar{Jel\,"ol\,'es}}}%
7177 \renewcommand*{\pagelistname}{\textmagyar{Oldalsz\,'am}}}%
7178 \renewcommand*{\glssymbolsgroupname}{\textmagyar{Jelek}}}%
7179 \renewcommand*{\glsnumbersgroupname}{\textmagyar{Sz\,'amjegyek}}}%
7180 }%
7181 \edef\captionsmagyar{\the\toks@}%
7182 }

Polish
7183 \@ifundefined{captionspolish}{}{%
7184 \expandafter\toks@\expandafter{\captionspolish
7185 \renewcommand*{\glossaryname}{\textpolish{S{\l}ownik termin\,'ow}}}%
7186 \renewcommand*{\acronymname}{\textpolish{Skr\,'ot}}}%
7187 \renewcommand*{\entryname}{\textpolish{Termin}}}%
7188 \renewcommand*{\descriptionname}{\textpolish{Opis}}}%
7189 \renewcommand*{\symbolname}{\textpolish{Symbol}}}%
7190 \renewcommand*{\pagelistname}{\textpolish{Strony}}}%
7191 \renewcommand*{\glssymbolsgroupname}{\textpolish{Symbole}}}%

```

```

7192 \renewcommand*{\glnumbersgroupname}{\textpolish{Liczby}}%
7193 }%
7194 \edef\captionspolish{\the\toks@}%
7195 }

```

### Portugues

```

7196 \@ifundefined{captionoportuges}{}{%
7197 \expandafter\toks@\expandafter{\captionoportuges
7198 \renewcommand*{\glossaryname}{\textportuges{Gloss\'ario}}%
7199 \renewcommand*{\acronymname}{\textportuges{Siglas}}%
7200 \renewcommand*{\entryname}{\textportuges{Nota\c c\~ao}}%
7201 \renewcommand*{\descriptionname}{\textportuges{Descri\c c\~ao}}%
7202 \renewcommand*{\symbolname}{\textportuges{S\'imbolo}}%
7203 \renewcommand*{\pagelistname}{\textportuges{Lista de P\'aginas}}%
7204 \renewcommand*{\glssymbolsgroupname}{\textportuges{S\'imbolos}}%
7205 \renewcommand*{\glnumbersgroupname}{\textportuges{N\'umeros}}%
7206 }%
7207 \edef\captionoportuges{\the\toks@}%
7208 }

```

## 6.3 Brazilian Dictionary

This is a dictionary file provided by Thiago de Melo for use with the package.

```

7209 \ProvidesDictionary{glossaries-dictionary}{Brazilian}

```

Provide Brazilian translations:

```

7210 \providetranslation{Glossary}{Gloss\'ario}
7211 \providetranslation{Acronyms}{Siglas}
7212 \providetranslation{Notation (glossaries)}{Nota\c c\~ao}
7213 \providetranslation{Description (glossaries)}{Descri\c c\~ao}
7214 \providetranslation{Symbol (glossaries)}{S\'imbolo}
7215 \providetranslation{Page List (glossaries)}{Lista de P\'aginas}
7216 \providetranslation{Symbols (glossaries)}{S\'imbolos}
7217 \providetranslation{Numbers (glossaries)}{N\'umeros}

```

## 6.4 Danish Dictionary

This is a dictionary file provided for use with the package.

```

7218 \ProvidesDictionary{glossaries-dictionary}{Danish}

```

Provide Danish translations:

```

7219 \providetranslation{Glossary}{Ordliste}
7220 \providetranslation{Acronyms}{Akronymer}
7221 \providetranslation{Notation (glossaries)}{Symbolforklaring}
7222 \providetranslation{Description (glossaries)}{Beskrivelse}
7223 \providetranslation{Symbol (glossaries)}{Symbol}
7224 \providetranslation{Page List (glossaries)}{Side}
7225 \providetranslation{Symbols (glossaries)}{Symboler}
7226 \providetranslation{Numbers (glossaries)}{Tal}

```

## 6.5 Dutch Dictionary

This is a dictionary file provided for use with the package.

```
7227 \ProvidesDictionary{glossaries-dictionary}{Dutch}
```

Provide Dutch translations:

```
7228 \providetranslation{Glossary}{Woordenlijst}
7229 \providetranslation{Acronyms}{Acroniemen}
7230 \providetranslation{Notation (glossaries)}{Benaming}
7231 \providetranslation{Description (glossaries)}{Beschrijving}
7232 \providetranslation{Symbol (glossaries)}{Symbool}
7233 \providetranslation{Page List (glossaries)}{Pagina's}
7234 \providetranslation{Symbols (glossaries)}{Symbolen}
7235 \providetranslation{Numbers (glossaries)}{Cijfers}
```

## 6.6 English Dictionary

This is a dictionary file provided for use with the package.

```
7236 \ProvidesDictionary{glossaries-dictionary}{English}
```

Provide English translations:

```
7237 \providetranslation{Glossary}{Glossary}
7238 \providetranslation{Acronyms}{Acronyms}
7239 \providetranslation{Notation (glossaries)}{Notation}
7240 \providetranslation{Description (glossaries)}{Description}
7241 \providetranslation{Symbol (glossaries)}{Symbol}
7242 \providetranslation{Page List (glossaries)}{Page List}
7243 \providetranslation{Symbols (glossaries)}{Symbols}
7244 \providetranslation{Numbers (glossaries)}{Numbers}
```

## 6.7 French Dictionary

This is a dictionary file provided for use with the package.

```
7245 \ProvidesDictionary{glossaries-dictionary}{French}
```

Provide French translations:

```
7246 \providetranslation{Glossary}{Glossaire}
7247 \providetranslation{Acronyms}{Acronymes}
7248 \providetranslation{Notation (glossaries)}{Terme}
7249 \providetranslation{Description (glossaries)}{Description}
7250 \providetranslation{Symbol (glossaries)}{Symbole}
7251 \providetranslation{Page List (glossaries)}{Pages}
7252 \providetranslation{Symbols (glossaries)}{Symboles}
7253 \providetranslation{Numbers (glossaries)}{Nombres}
```

## 6.8 German Dictionary

This is a dictionary file provided for use with the package.

```
7254 \ProvidesDictionary{glossaries-dictionary}{German}
```

Provide German translations (quite a few variations were suggested for German; I settled on the following):

```
7255 \providetranslation{Glossary}{Glossar}
7256 \providetranslation{Acronyms}{Akronyme}
7257 \providetranslation{Notation (glossaries)}{Bezeichnung}
7258 \providetranslation{Description (glossaries)}{Beschreibung}
7259 \providetranslation{Symbol (glossaries)}{Symbol}
7260 \providetranslation{Page List (glossaries)}{Seiten}
7261 \providetranslation{Symbols (glossaries)}{Symbole}
7262 \providetranslation{Numbers (glossaries)}{Zahlen}
```

## 6.9 Irish Dictionary

This is a dictionary file provided for use with the package.

```
7263 \ProvidesDictionary{glossaries-dictionary}{Irish}
```

Provide Irish translations:

```
7264 \providetranslation{Glossary}{Gluais}
7265 \providetranslation{Acronyms}{Acrainmneacha}
7266 \providetranslation{Notation (glossaries)}{Ciall}
7267 \providetranslation{Description (glossaries)}{Tuairisc}
7268 \providetranslation{Symbol (glossaries)}{Comhartha}
7269 \providetranslation{Page List (glossaries)}{Leathanaigh}
7270 \providetranslation{Symbols (glossaries)}{Comhartha'\{i}}
7271 \providetranslation{Numbers (glossaries)}{Uimhreacha}
```

## 6.10 Italian Dictionary

This is a dictionary file provided for use with the package.

```
7272 \ProvidesDictionary{glossaries-dictionary}{Italian}
```

Provide Italian translations:

```
7273 \providetranslation{Glossary}{Glossario}
7274 \providetranslation{Acronyms}{Acronimi}
7275 \providetranslation{Notation (glossaries)}{Nomenclatura}
7276 \providetranslation{Description (glossaries)}{Descrizione}
7277 \providetranslation{Symbol (glossaries)}{Simbolo}
7278 \providetranslation{Page List (glossaries)}{Elenco delle pagine}
7279 \providetranslation{Symbols (glossaries)}{Simboli}
7280 \providetranslation{Numbers (glossaries)}{Numeri}
```

## 6.11 Magyar Dictionary

This is a dictionary file provided for use with the package.

```
7281 \ProvidesDictionary{glossaries-dictionary}{Magyar}
```

Provide translations:

```
7282 \providetranslation{Glossary}{Sz\'ojegy\'ek}
7283 \providetranslation{Acronyms}{Bet\'H uszavak}
```

```

7284 \providetranslation{Notation (glossaries)}{Kifejez\`es}
7285 \providetranslation{Description (glossaries)}{Magyar\`azat}
7286 \providetranslation{Symbol (glossaries)}{Jel\`ol\`es}
7287 \providetranslation{Page List (glossaries)}{Oldalsz\`am}
7288 \providetranslation{Symbols (glossaries)}{Jelek}
7289 \providetranslation{Numbers (glossaries)}{Sz\`amjegyek}

```

## 6.12 Polish Dictionary

This is a dictionary file provided for use with the package.

```
7290 \ProvidesDictionary{glossaries-dictionary}{Polish}
```

Provide Polish translations:

```

7291 \providetranslation{Glossary}{S{\l}ownik termin\`ow}
7292 \providetranslation{Acronyms}{Skr\`ot}
7293 \providetranslation{Notation (glossaries)}{Termin}
7294 \providetranslation{Description (glossaries)}{Opis}
7295 \providetranslation{Symbol (glossaries)}{Symbol}
7296 \providetranslation{Page List (glossaries)}{Strony}
7297 \providetranslation{Symbols (glossaries)}{Symbole}
7298 \providetranslation{Numbers (glossaries)}{Liczby}

```

## 6.13 Serbian Dictionary

This dictionary was provided by Zoran Filipovic.

```

7299 \ProvidesDictionary{glossaries-dictionary}{Serbian}
7300 \providetranslation{Glossary}{Mali re\`v cnik}
7301 \providetranslation{Acronyms}{Skra\` cenice}
7302 \providetranslation{Notation (glossaries)}{Oznaka}
7303 \providetranslation{Description (glossaries)}{Opis}
7304 \providetranslation{Symbol (glossaries)}{Simbol}
7305 \providetranslation{Page List (glossaries)}{Stranica}
7306 \providetranslation{Symbols (glossaries)}{Simboli}
7307 \providetranslation{Numbers (glossaries)}{Brojevi}

```

## 6.14 Spanish Dictionary

This is a dictionary file provided for use with the package.

```
7308 \ProvidesDictionary{glossaries-dictionary}{Spanish}
```

Provide Spanish translations:

```

7309 \providetranslation{Glossary}{Glosario}
7310 \providetranslation{Acronyms}{Siglas}
7311 \providetranslation{Notation (glossaries)}{Entrada}
7312 \providetranslation{Description (glossaries)}{Descripci\`on}
7313 \providetranslation{Symbol (glossaries)}{S\`{\i}mbolo}
7314 \providetranslation{Page List (glossaries)}{Lista de p\`aginas}
7315 \providetranslation{Symbols (glossaries)}{S\`{\i}mbolos}
7316 \providetranslation{Numbers (glossaries)}{N\`umeros}

```

## Glossary

`makeindex` An indexing application. 17

`xindy` An flexible indexing application with multilingual support written in Perl. 17

## Change History

- 1.01
  - General: Added range facility in format key ..... 60
  - `\writeist`: Added spaces after `'delimN` and `'delimR` in ist file 117
- 1.03
  - `\makefirstuc`: changed `'protected@edef` to `'def` ..... 170
- 1.04
  - General: Added `'glstextformat` ... 56
- 1.05
  - `\glossarysection`: added `'@mk-` both to `'glossarysection` .... 26
  - `\glsmakefirstuc`: new ..... 171
  - `\newglossaryentry`: Changed the default value of the sort key to just the value of the name key ..... 47
- 1.06
  - General: now requires `etoolbox` . 170
  - `\capitalisewords`: new ..... 171
  - `\xcapitalisewords`: new .... 172
- 1.07
  - `\@gls@link`: fixed bug caused by `\theglsentrycounter` setting the page number too soon 59
  - `\glsadd`: fixed bug caused by `\theglsentrycounter` setting the page number too soon ..... 115
- 1.08
  - General: Added babel support ... 21
  - `listgroup`: changed `listgroup` style to use `\glsgetgrouptitle` ..... 177
  - `altlistgroup`: changed `altlistgroup` style to use `\glsgetgrouptitle` ..... 178
- 1.1
  - `\newglossaryentry`: Fixed error message to say “description key” rather than “desc key” .. 47
- 1.1
  - `\@glossarysection`: numbered sections and auto label added 27
  - `\@gls@tmpb`: changed `\toksdef` to `\newtoks` ..... 63
  - `\@gls@toc`: numberline added .. 28
  - `\@p@glossarysection`: numbered sections and auto label added ..... 27
  - General: Added support for translator package ..... 21
  - `amsgen` now loaded (`\new@ifnextchar` needed) ..... 3
  - `translate`: `translate` option added ..... 15
  - `\setglossarysection`: new ... 27
  - `numberedsection`: `numberedsection` package option added . 6
  - `numberline`: `numberline` option added ..... 5
- 1.12
  - `\@GLSpl`: now uses `'glsentrydescplural` and `'glsentrysymbolplural` instead of `'glsentrydesc` and `'glsentrysymbol` ..... 77
  - `\@Glspl@`: now uses `'glsentrydescplural` and `'glsentrysymbolplural` instead of `'glsentrydesc` and `'glsentrysymbol` ..... 75
  - `\@glspl@`: now uses `'glsentrydescplural` and `'glsentrysymbolplural` instead of `'glsentrydesc` and `'glsentrysymbol` ..... 74

General: added check for 'hypertarget separate to 'hyperlink (memoir defines 'hyperlink but not 'hypertarget) ...	68	fixed bug ('GLSdesc shouldn't use 'gls@⟨type⟩@display) ....	88	fixed bug ('GLSdesc shouldn't use 'gls@⟨type⟩@display) ....	87	fixed bug ('glsdesc shouldn't use 'gls@⟨type⟩@display) .....	87	fixed bug ('GLSfirst shouldn't use 'gls@⟨type⟩@display) ....	82	fixed bug ('Glsfirst shouldn't use 'gls@⟨type⟩@display) .....	81	fixed bug ('glsfirst shouldn't use 'gls@⟨type⟩@display) .....	81	fixed bug ('GLSfirstplural shouldn't use 'gls@⟨type⟩@display) .....	85	fixed bug ('Glsfirstplural shouldn't use 'gls@⟨type⟩@display) .....	84	fixed bug ('glsfirstplural shouldn't use 'gls@⟨type⟩@display) .....	84	fixed bug ('GLSname shouldn't use 'gls@⟨type⟩@display) ....	86	fixed bug ('glsname shouldn't use 'gls@⟨type⟩@display) ..	85, 86	fixed bug ('GLSplural shouldn't use 'gls@⟨type⟩@display) ....	83	fixed bug ('Glsplural shouldn't use 'gls@⟨type⟩@display) ....	83	fixed bug ('glsplural shouldn't use 'gls@⟨type⟩@display) ....	82	fixed bug ('GLSsymbol shouldn't use 'gls@⟨type⟩@display) .....	91	fixed bug ('Glsymbol shouldn't use 'gls@⟨type⟩@display) ....	90	fixed bug ('glsymbol shouldn't use 'gls@⟨type⟩@display) ....	90	fixed bug ('glssymbolplural shouldn't use 'gls@⟨type⟩@display) .....	91	fixed bug ('GLStext shouldn't use 'gls@⟨type⟩@display) ....	80	fixed bug ('Glstext shouldn't use 'gls@⟨type⟩@display) .....	80	fixed bug ('glstext shouldn't use 'gls@⟨type⟩@display) .....	80	descriptionplural: new .....	43	\Glsentrydescplural: New ..	110	\glsentrydescplural: New ..	110	\Glsentrysymbolplural: New	111	\glsentrysymbolplural: New	110	\newglossaryentry: Changed default first plural to be first key with s appended (was text key with s appended) .....	47	descriptionplural support added .....	47	symbolplural support added ..	47	\SetDescriptionFootnoteAcronymStyle: Added 'protect before 'footnote and 'glslink .....	153	\SetFootnoteAcronymStyle: Added 'protect before 'footnote and 'glslink .....	157	symbolplural: new .....	44	1.13	General: Add Polish support	247, 250	fixed bug that ignores 3rd parameter .....	80–93	\ACRfullpl: new .....	149	\Acrfullpl: new .....	149	\acrfullpl: new .....	148	\acrpluralsuffix: New .....	146	\newacronym: Removed restriction on only using 'newacronym in the preamble	146	\newglossaryentry: Changed default first value .....	47	Changed default firstplural value .....	47	Removed restriction on only using 'newglossaryentry in the preamble .....	51	1.14	\@gls@hypergroup: new .....	173	General: added nonnumberlist key to 'printglossary .....	136	added numberedsection key to 'printglossary .....	136	\firstacronymfont: new ....	149	\glsautoprefix: new .....	6	\glsnavhyperlink: changed 'edef to 'protected@edef ...	172
---	----	---	----	---	----	--	----	--	----	---	----	---	----	---	----	---	----	---	----	---	----	---	--------	---	----	---	----	---	----	--	----	--	----	--	----	--	----	---	----	--	----	--	----	------------------------------	----	-----------------------------	-----	-----------------------------	-----	----------------------------	-----	----------------------------	-----	--	----	---------------------------------------	----	-------------------------------	----	---	-----	--	-----	-------------------------	----	------	-----------------------------	----------	--	-------	-----------------------	-----	-----------------------	-----	-----------------------	-----	-----------------------------	-----	--	-----	--	----	---	----	---	----	------	-----------------------------	-----	--	-----	---	-----	-----------------------------	-----	---------------------------	---	--	-----

<code>\glsnavhypertarget:</code>	added	checking if footnote option has been used	74
	write to aux file		172
<code>\glsnavigation:</code>	changed to only use labels for groups that are present	<code>\@glstarget:</code> raised the hypertarget so the target text doesn't scroll off the top of the page	68
1.15		<code>\newglossaryentry:</code> Changed def to let	47
<code>\@gls@link:</code>	added 'glslabel	Changed if to ifx	49
General:	Added 'glssettoctitle		
<code>\gls@hypergroup</code>	new		173
<code>\glsnavhypertarget:</code>	added		
	check if rerun required		172
<code>\glssettoctitle:</code>	new		20
<code>\newglossaryentry:</code>	check for '@glo@first in description		50
	check for '@glo@text in symbol		51
<code>\printglossary:</code>	changed the way the TOC title is set		132
1.16			
<code>\@GLS@:</code>	Test glossary type is 'acronymtype in addition to checking if footnote option has been used		73, 231
<code>\@GLSpl:</code>	Test glossary type is 'acronymtype in addition to checking if footnote option has been used		77
<code>\@GLS@:</code>	Test glossary type is 'acronymtype in addition to checking if footnote option has been used		72
<code>\@GLspl@:</code>	Test glossary type is 'acronymtype in addition to checking if footnote option has been used		76
<code>\@gls@:</code>	Test glossary type is 'acronymtype in addition to checking if footnote option has been used		70
<code>\@gls@pl@:</code>	Test glossary type is 'acronymtype in addition to checking if footnote option has been used		232
<code>\@glsdisp:</code>	Test glossary type is 'acronymtype in addition to checking if footnote option has been used		79
<code>\@glspl@:</code>	Test glossary type is 'acronymtype in addition to		
1.17			
<code>\@do@wrglossary:</code>	new		128
<code>\@do@seeglossary:</code>	new		130
<code>\@glo@storeentry:</code>	new		52
<code>\@glossary:</code>	changed definition to use \index instead of \@index		126
<code>\@glsdefaultplural:</code>	new		46
<code>\@glsdefaultsort:</code>	new		46
<code>\@glshypernumber:</code>	new		143
<code>\@glsnoname:</code>	new		46
<code>\@glsnonextpages:</code>	new		136
<code>\@wrglossary:</code>	modified to allow for xindy support		127
General:	added Brazilian dictionary		251
	Added Brazilian support		247
	added xindy support		17
parent:	new		44
see:	new		44
<code>\gls@suffixF:</code>	new		24
<code>\gls@suffixFF:</code>	new		24
<code>\glshyperlink:</code>	new		115
<code>\glshypernumber:</code>	modified to allow material to be attached to location		143
<code>\glsnavhyperlink:</code>	replaced 'hyperlink to '@glslink		172
<code>\glsnavhypertarget:</code>	replaced 'hypertarget to '@glstarget		172
<code>\glssee:</code>	new		131
<code>\glsseeformat:</code>	new		131
<code>\glsSetSuffixF:</code>	new		24
<code>\glsSetSuffixFF:</code>	new		25
<code>\ifglsxindy:</code>	new		17
<code>\istfilename:</code>	added xindy support		23
<code>\newglossaryentry:</code>	added non-umberlist key		47
	added parent key		47
	added see key		47

Stored main part of entry format	1.2	General: fixed bug in ngerman	
when entry is defined . . . . .	51	captions . . . . .	244
\newglossarystyle: made 'new-		2.01	
glossarystyle long . . . . .	142	\@gls@link: moved \@do@wrglossary	
\nopostdesc: new . . . . .	23	before term is displayed to pre-	
nonumberlist: new . . . . .	44	vent unwanted whatsit . . . . .	59
\printglossary: added check to		General: added nomain package	
determine if 'printglossary is		option . . . . .	11
already defined . . . . .	132	\forall glossaries: replaced	
added print language to aux file	132	\ifthenelse with \ifx . . . . .	37
order: order package option		\forall glossentries: replaced	
added . . . . .	17	\ifthenelse with \ifx . . . . .	38
\writeist: added xindy support	117	\glsdefmain: new . . . . .	11
1.18		\glsdescwidth: changed	
\@gls@loadlist: new . . . . .	7	\linewidth to \hsize . 179, 194	
\@gls@loadlong: new . . . . .	7	\glslistdottedwidth: changed	
\@gls@loadsuper: new . . . . .	7	\linewidth to \hsize . . . . .	179
\@gls@loadtree: new . . . . .	8	\glsPagelistwidth: changed	
\glsTarget: new . . . . .	139	\linewidth to \hsize . 180, 194	
\newglossaryentry: Changed		\writeist: removed item_02 - no	
default value of sort to '@gls-		such makeindex key . . . . .	121
defaultsort . . . . .	47	2.02	
moved sort sanitization to 'new-		General: Changed Brazil to Brazil-	
glossaryentry . . . . .	51	ian . . . . .	251
\oldacronym: new . . . . .	146	false will prevent automatic	
nolist: new . . . . .	7	loading of translator package	18
nolong: new . . . . .	7	\glossarymark: New . . . . .	26
sort: moved sanitization to 'new-		\glossarysection: changed	
glossaryentry . . . . .	43	'@mkboth to 'glossarymark . .	26
nostyles: new . . . . .	8	\printglossary: suppressed	
nosuper: new . . . . .	7	warning globally rather than	
notree: new . . . . .	8	locally . . . . .	135
1.19		2.03	
\glsClearpage: new . . . . .	28	\@GLS@: Added check for hyper-	
\glsdisp: new . . . . .	78	first . . . . .	73
\SetDescriptionAcronymStyle:		\@GLSpl: Added check for hyper-	
changed 'acronymfont to use		first . . . . .	77
'textsmaller instead of 'smaller	156	\@Gls@: Added check for hyper-	
\SetDescriptionFootnoteAcronymStyle		first . . . . .	72
changed 'acronymfont to use		\@GLspl@: Added check for hyper-	
'textsmaller instead of 'smaller	153	first . . . . .	76
\SetFootnoteAcronymStyle:		\@gls@: Added check for hyper-	
changed 'acronymfont to use		first . . . . .	70
'textsmaller instead of 'smaller	157	\@gls@link: new . . . . .	58
\SetSmallAcronymStyle:		\@gls@link: added \leavevmode	
changed 'acronymfont to use		. . . . .	59
'textsmaller instead of 'smaller	159	Moved entry existence check to	
		avoid duplicate code . . . . .	59

\@glsdisp: Added check for hyperfirst .....	79	\glsentryuseriv:new .....	112
\@glspl@: Added check for hyperfirst .....	74	\Glsentryuseriv:new .....	112
\glossarymark: Added check to see if it's already defined ....	26	\glsentryuseriv:new .....	112
hyperfirst:new .....	16	\glsentryuseriv:new .....	112
2.04		\newglossary: added check to determine if \gls@<type>@display and \gls@<type>@displayfirst have been defined. ....	41
\@GLS@: Changed test to check if glossary type has been identified as a list of acronyms ....	73	\newglossaryentry: added user1-6 keys .....	47
\@GLSpl@: Changed test to check if glossary type has been identified as a list of acronyms ....	77	\SetAcronymLists:new .....	13
\@GLS@: Changed test to check if glossary type has been identified as a list of acronyms ....	72	\SetDefaultAcronymDisplayStyle:new .....	150
\@GLSpl@: Changed test to check if glossary type has been identified as a list of acronyms ...	76	\SetDefaultAcronymStyle:new .....	151
\@glossaryentryfield:new ..	52	\SetDescriptionAcronymDisplayStyle:new .....	154
\@glossarysubentryfield:new .....	52	\SetDescriptionDUAAcronymDisplayStyle:new .....	153
\@gls@: Changed test to check if glossary type has been identified as a list of acronyms ....	70	\SetDescriptionFootnoteAcronymDisplayStyle:new .....	151
\@glsacronymlists:new .....	12	\SetDUADisplayStyle:new ..	159
\@glsdisp: Changed test to check if glossary type has been identified as a list of acronyms ...	79	\SetFootnoteAcronymDisplayStyle:new .....	156
\@glspl@: Changed test to check if glossary type has been identified as a list of acronyms ...	74	\SetSmallAcronymDisplayStyle:new .....	157
\@newglossaryentryposthook:new .....	51	2.05	
\@newglossaryentryprehook:new .....	51	\@glsdisp: Added closing brace. Patch provided by Sergiu Dotenco .....	78
acronymlists:new .....	13	Removed spurious brace. Patch provided by Sergiu Dotenco ..	79
\DeclareAcronymList:new ...	12	\writeist: Added \string before opening and closing braces. Patch provided by Segiu Dotenco .....	122
\DefineAcronymSynonyms:new	162	2.06	
\glsadd: fixed bug that ignored counter .....	115	\altnewglossary:new .....	41
\Glsentryuseri:new .....	111	\CustomAcronymFields:new ..	161
\Glsentryuseri:new .....	111	\CustomNewAcronymDef:new ..	161
\Glsentryuserii:new .....	112	\SetCustomDisplayStyle:new	161
\Glsentryuserii:new .....	112	\SetCustomStyle:new .....	162
\Glsentryuseriii:new .....	112	2.07	
\Glsentryuseriii:new .....	112	General: glssadd format key stored in \@glsnumberformat (was mistakenly stored in \@glo@format) .....	115
\Glsentryuseriv:new .....	112		

### 3.0

\@do@wrglossary: added check for hyper location prefix ...	129	\addglossarytocaptions: replaced \@ifundefined with \ifcsundef .....	21
modified to use new format ..	128	savewrites: new .....	18
\@glossarysec: replaced \@ifundefined with \ifcsundef .....	5	see: added \@glo@seeautonumberlist .....	44
\@do@seeglossary: Sanitize and escape cross-referencing information .....	130	seeautonumberlist: new .....	7
\@gls@counterwithin: new ....	9	\glossarymark: replaced \@ifundefined with \ifcsundef .....	26
\@gls@ifinlist: new .....	29	\glossarysection: replaced \@ifundefined with \ifcsundef .....	26
\@gls@link: added \@gls@saveentrycounter .....	59	\glossarystyle: replaced \@ifundefined with \ifcsundef .....	141
added \@gls@setsort .....	59	\gls@codepage: replaced \@ifundefined with \ifcsundef .....	17
\@gls@saveentrycounter: new	59	\gls@doclearpage: replaced \@ifundefined with \ifcsundef .....	28
\@gls@setupsort@def: new ...	10	\glsadd: added \@gls@saveentrycounter .....	116
\@gls@setupsort@standard: new .....	9	\GlsAddXdyCounters: new ....	29
\@gls@setupsort@use: new ...	10	\glentrycounterlabel: new	138
\@gls@xdy@locationlist: new	32	\glentryitem: new .....	139
\@glslink: replaced \@ifundefined with \ifcsundef .....	68	\Glsentrylong: new .....	113
\@glsnextpages: new .....	136	\glentrylong: new .....	113
\@makeglossary: Added check for savewrites .....	123	\Glsentrylongpl: new .....	113
\@set@glo@numformat: added 4th argument .....	61	\glentrylongpl: new .....	113
\@wrglossary: modified to take into account savewrites ....	127	\Glsentryshort: new .....	113
\@xdyattributelist: new ....	29	\glentryshort: new .....	112
General: added prefix to hyperlink .....	144	\Glsentryshortpl: new .....	113
etoolbox now loaded .....	4	\glentryshortpl: new .....	113
replaced \@ifundefined with \ifcsundef .....	19, 57, 135	\glsgroupptitle: replaced \@ifundefined with \ifcsundef .....	141
\acrfootnote: new .....	151	\glshyperlink: changed default from \glentryname to \glentrytext .....	115
\ACRfull: added starred version	148	\glshypernumber: replaced \@ifundefined with \ifcsundef .....	143
\Acrfull: added starred version	147	\glsnumberformat: replaced \@ifundefined with \ifcsundef .....	25
\acrfullpl: added starred version .....	149	\glsrefentry: new .....	138
\Acrfullpl: added starred version .....	149		
\acrfullpl: added starred version .....	148		
\acrlinkfootnote: new ....	151		
\acrnolinkfootnote: new ...	151		

<code>\glsresetsubentrycounter:</code>	<code>counter:replaced \@ifundefined</code>
new ..... 137	with <code>\ifcsundef</code> ..... 44
<code>\glsseeitem:</code> hyperlink uses	<code>\printglossary:</code> added
<code>\glsseeitemformat</code> instead	<code>\currentglossary</code> ..... 133
of <code>\glsentryname</code> ..... 132	added <code>\glsnextpages</code> ..... 133
<code>\glsseeitemformat:new</code> .... 132	make toctitle default to title .. 133
<code>\gls sortnumberfmt:new</code> ..... 9	replaced <code>\@ifundefined</code> with
<code>\glsstepentry:new</code> ..... 138	<code>\ifcsundef</code> ..... 132, 134
<code>\glsstepsubentry:new</code> ..... 138	<code>\SetDescriptionFootnoteAcronymDisplayStyle:</code>
<code>\gls subentrycounterlabel:</code>	expanded options link op-
new ..... 138	tions ..... 151
<code>\gls subentryitem:new</code> ..... 139	<code>\setentrycounter:</code> added op-
<code>theglossary:replaced \@ifundefined</code>	tional argument ..... 141
with <code>\ifcsundef</code> ..... 139	<code>\showacronymlists:new</code> .... 168
<code>short:new</code> ..... 45	<code>\showglocounter:new</code> ..... 166
<code>shortplural:new</code> ..... 45	<code>\showgloDESC:new</code> ..... 167
<code>\ifglossaryexists:</code> re-	<code>\showgloDESCplural:new</code> ... 167
placed <code>\@ifundefined</code> with	<code>\showglofirst:new</code> ..... 165
<code>\ifcsundef</code> ..... 38	<code>\showglofirstpl:new</code> ..... 165
<code>\ifglsentryexists:</code> re-	<code>\showgloflag:new</code> ..... 168
placed <code>\@ifundefined</code> with	<code>\showgloindex:new</code> ..... 168
<code>\ifcsundef</code> ..... 38	<code>\showglolevel:new</code> ..... 165
<code>\istfile:deprecated</code> ..... 125	<code>\showgloNAME:new</code> ..... 167
<code>glossaryentry:new</code> ..... 137	<code>\showgloparent:new</code> ..... 165
<code>glossarysubentry:new</code> ..... 137	<code>\showgloplural:new</code> ..... 165
<code>\newglossary:added \@gls@defsortcount</code>	<code>\showglosort:new</code> ..... 167
..... 41	<code>\showglossaries:new</code> ..... 168
replaced <code>\@ifundefined</code> with	<code>\showglossarycounter:new</code> . 169
<code>\ifcsundef</code> ..... 41	<code>\showglossaryentries:new</code> . 169
<code>\newglossaryentry:</code> added	<code>\showglossaryin:new</code> ..... 168
<code>\@gls@defsort</code> ..... 51	<code>\showglossaryout:new</code> ..... 168
added short and long keys .... 48	<code>\showglossarytitle:new</code> ... 169
replaced <code>\@ifundefined</code> with	<code>\showglosymbol:new</code> ..... 167
<code>\ifcsundef</code> ..... 48	<code>\showglosymbolplural:new</code> . 167
replaced <code>\DeclareRobustCommand</code>	<code>\showglotext:new</code> ..... 165
with <code>\newrobustcmd</code> ..... 46	<code>\showglotype:new</code> ..... 165
<code>\newglossarystyle:</code> re-	<code>\showglouserI:new</code> ..... 166
placed <code>\@ifundefined</code> with	<code>\showglouserII:new</code> ..... 166
<code>\ifcsundef</code> ..... 142	<code>\showglouserIII:new</code> ..... 166
<code>entrycounter:new</code> ..... 8	<code>\showglouserIV:new</code> ..... 166
<code>entrycounterwithin:new</code> ..... 8	<code>\showglouserv:new</code> ..... 166
<code>\oldacronym:replaced \@ifundefined</code>	<code>\showglouservi:new</code> ..... 167
with <code>\ifcsundef</code> ..... 146	<code>subentrycounter:new</code> ..... 9
<code>compatible-2.07: compatible-</code>	<code>\writeist:</code> added xindy-only
2.07 option added ..... 18	macro definitions to glossary
<code>long:new</code> ..... 46	open tag ..... 119
<code>longplural:new</code> ..... 46	modified to support new for-
<code>nonumberlist:now boolean</code> ... 44	mat ..... 117
<code>sort:new</code> ..... 9	

3.01		
General: made robust	72	\Glsymbolplural: made robust 92
\ACRfull: made robust	148	\glssymbolplural: made robust 91
\Acrfull: made robust	147	\Glstext: made robust 80
\acrfull: made robust	147	\glstext: made robust 79
\acrfullformat:	removed	\GLSuseri: made robust 94
\acronymfont as it should already be set in the second argument.	147	\Glsuseri: made robust 93
\ACRfullpl: made robust	149	\glsuseri: made robust 93
\Acrfullpl: made robust	149	\GLSuserii: made robust 95
\acrfullpl: made robust	148	\Glsuserii: made robust 95
\ACRlong: made robust	106	\glsuserii: made robust 94
\Acrlong: made robust	106	\GLSuseriii: made robust 97
\acrlong: made robust	105	\Glsuseriii: made robust 96
\ACRlongpl: made robust	108	\glsuseriii: made robust 96
\Acrlongpl: made robust	108	\GLSuseriv: made robust 98
\acrlongpl: made robust	107	\Glsuseriv: made robust 98
\ACRshort: made robust	103	\glsuseriv: made robust 97
\Acrshort: made robust	102	\GLSuserv: made robust 100
\acrshort: made robust	102	\Glsuserv: made robust 99
\ACRshortpl: made robust	105	\glsuserv: made robust 99
\Acrshortpl: made robust	104	\GLSuservi: made robust 101
\acrshortpl: made robust	103	\Glsuservi: made robust 101
\Gls: made robust	71	\glsuservi: made robust 100
\glsadd: made robust	115	\glswritefiles: added check for empty glossaries 125
\glsaddall: made robust	116	3.02
\GLSdesc: made robust	88	\@@do@wrglossary: changed
\Glsdesc: made robust	87	\@glslocref to \theglentrycounter 129
\glsdesc: made robust	87	\@do@wrglossary: changed
\GLSdescplural: made robust	89	\@do@wr@glossary to test for indexonlyfirst option; put old
\Glsdescplural: made robust	89	\@do@wr@glossary code into
\glsdescplural: made robust	88	\@@do@wrglossary 127
\glsfirst: made robust	81	\@gls@missingnumberlist:
\GLSfirstplural: made robust	85	new 46
\Glsfirstplural: made robust	84	\@wrglossary: added check for glossary file defined 127
\glsfirstplural: made robust	84	General: added check for polyglossia 19
\glslink: made robust	58	reversed order of package check 22
\GLSname: made robust	86	savenumberlist: new 7
\Glsname: made robust	86	ucmark: new 8
\glsname: made robust	85	\gls@save@numberlist: new 132
\GLSpl: made robust	77	\glsdisplaynumberlist: new 114
\Glspl: made robust	75	\glentrycounter: set default value 59
\glspl: made robust	74	\Glsentryfull: fixed bug (replaced \glentryshortpl with \glentryshort) 113
\GLSplural: made robust	83	
\GLSsymbol: made robust	91	
\Glsymbol: made robust	90	
\glssymbol: made robust	90	
\GLSsymbolplural: made robust	92	

<code>\glsentryfullpl</code> : fixed bug (replaced <code>\glsentryshort</code> with <code>\glsentryshortpl</code> ) .....	113	<code>longragged3col</code> : added check for <code>glsnogroupskip</code> .....	187
<code>\glsentrynumberlist</code> : new ..	114	<code>nopostdot</code> : new .....	8
<code>\glsmoveentry</code> : new .....	52	<code>\printglossary</code> : allow title to override default <code>toctitle</code> ....	133
<code>\glsnumlistlastsep</code> : new ...	115	<code>tree</code> : added check for <code>glsnogroupskip</code> .....	208
<code>\glsnumlistsep</code> : new .....	115	<code>treenoname</code> : added check for <code>glsnogroupskip</code> .....	210
<code>\glsresetsubentrycounter</code> : new .....	137	<code>super</code> : added check for <code>glsnogroupskip</code> .....	195
<code>\glswritefiles</code> : added check for existence of token in case <code>\makeglossaries</code> has been omitted .....	125	<code>super3col</code> : added check for <code>glsnogroupskip</code> .....	196
<code>\ifglshaschildren</code> : new .....	39	<code>super4col</code> : added check for <code>glsnogroupskip</code> .....	198
<code>\ifglshasparent</code> : new .....	39	<code>superragged</code> : added check for <code>glsnogroupskip</code> .....	201
<code>\makeglossaries</code> : added list parser .....	125	<code>superragged3col</code> : added check for <code>glsnogroupskip</code> .....	203
<code>indexonlyfirst</code> : new .....	16		
<code>\newglossaryentry</code> : added numberlist element .....	50		
<code>\printglossary</code> : add a way to fetch current entry label ...	134	<code>3.04</code>	
<code>\renewglossarystyle</code> : new ..	142	<code>\@@do@wrglossary</code> : changed <code>\theglsentrycounter</code> back to <code>\@glslocref</code> .....	129
<code>\showglossaryentries</code> : fixed misspelt command .....	169	modified to compensate for possible incorrect page number .....	128
<code>\SmallNewAcronymDef</code> : fixed broken short and long plural	158	<code>\@gls@escbsdq</code> : unsanitize <code>\gls@numberpage</code> , <code>\gls@alphpage</code> , <code>\gls@Alphpage</code> and <code>\gls@romanpage</code> .....	62
<b>3.03</b>		<b>General</b> : Added check for doc package .....	4
<code>\@gls@sanitizesort</code> : new ....	14	added <code>datatool-base</code> as a required package .....	3
<code>\@gls@setupsort@standard</code> : used <code>\@gls@sanitizesort</code> ..	9	added local key .....	58
<b>General</b> : allow title to set <code>toctitle</code>	135	<code>\changes</code> : new .....	4
<code>altlongragged4col</code> : added check for <code>glsnogroupskip</code> ..	189	<code>\gls@Alphpage</code> : new .....	128
<code>altsuperragged4col</code> : added check for <code>glsnogroupskip</code> ..	205	<code>\gls@alphpage</code> : new .....	128
<code>alttree</code> : added check for <code>glsnogroupskip</code> .....	213	<code>\gls@disablepagerefexpansion</code> : new .....	127
<code>index</code> : added check for <code>glsnogroupskip</code> .....	207	<code>\gls@numberpage</code> : new .....	128
<code>nogroupskip</code> : new .....	8	<code>\gls@protected@pagefmts</code> : new .....	127
<code>long</code> : added check for <code>glsnogroupskip</code> .....	180	<code>\gls@romanpage</code> : new .....	128
<code>long3col</code> : added check for <code>glsnogroupskip</code> .....	181	<code>\glsdefmain</code> : added check for doc package .....	11
<code>long4col</code> : added check for <code>glsnogroupskip</code> .....	183	<code>\glsorg@endtheglossary</code> : new .	5
<code>longragged</code> : added check for <code>glsnogroupskip</code> .....	186	<code>\glsorg@glossary</code> : new .....	4
		<code>\glsorg@theglossary</code> : new ....	5

\glsorg@wrglossary:new .....	4	mcolindex: replaced '2' with \glsmcols .....	191
altlist: replaced \newline with paragraph break .....	178	mcoltree: replaced '2' with \glsmcols .....	191
\PrintChanges:new .....	5	mcoltreenoname: replaced '2' with \glsmcols .....	192
\printglossary: Moved aux write to end of document to prevent unwanted whatsit oc- curring here. ....	134	\gls@protected@pagefmts: added Roman to list .....	127
3.05		\gls@Romanpage:new .....	128
\@@do@wrglossary: add Roman case. Fixed bugs in the else statements .....	128	\GlsDeclareNoHyperList:new	13
\@gls@link: added check for "no- hypertypes" .....	59	\glsgetgrouplabel: fixed bug (typo in \equal) .....	141
\@gls@nohyperlist:new .....	13	\nopostdesc: made robust .....	23
mcolalmtree: replaced '2' with \glsmcols .....	193	nohypertypes: new .....	14
		3.06	
		\@xdy@main@language: Changed back to using \languagename	17
		\findrootlanguage: Obsoleted	36

## Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in *roman* refer to the code lines where the entry is used.

<b>Symbols</b>	
\@do@wrglossary .....	128
\@glossarysec .....	5
\@glossaryseclabel .....	6
\@glossarysecstar .....	6
\@ACRlong .....	236
\@ACRshort .....	235
\@Acrlong .....	235
\@Acrshort .....	235
\@GLS@ .....	72, 230
\@GLSpl .....	77
\@GLSpl@ .....	233
\@Gls@ .....	71, 229
\@Glspl@ .....	75, 232
\@acrlong .....	235
\@acrshort .....	234
\@addtoacronymlists .....	12
\@delimN .....	144
\@delimR .....	143
\@disable@onlypremakeg .....	20
\@disable@premakecs .....	20
\@disabled@gladdxdycounters ..	30
\@do@seeglossary .....	130
\@do@wrglossary .....	127, 215
\@glo@seeautonumberlist .....	7
\@glo@storeentry .....	52
\@glo@types .....	40
\@glossary .....	126
\@glossary@default@style .....	6
\@glossaryentryfield .....	52, 236
\@glossarysection .....	27
\@glossarysubentryfield ..	52, 236
\@gls .....	70
\@gls@ .....	70, 228
\@gls@@link .....	58
\@gls@addpredefinedattributes	31
\@gls@checkactual .....	66
\@gls@checkbar .....	65
\@gls@checkescactual .....	64
\@gls@checkescbar .....	64
\@gls@checkesclevel .....	65
\@gls@checkescquote .....	63
\@gls@checklevel .....	66
\@gls@checkmkidxchars .....	62
\@gls@checkquote .....	63
\@gls@codepage .....	36
\@gls@counterwithin .....	9
\@gls@escbsdq .....	61
\@gls@fixbraces .....	131
\@gls@getcounter .....	42
\@gls@getcounterprefix .....	130
\@gls@hypergroup .....	173
\@gls@ifinlist .....	29
\@gls@link .....	59
\@gls@loadlist .....	7
\@gls@loadlong .....	7
\@gls@loadsuper .....	7
\@gls@loadtree .....	8
\@gls@missingnumberlist .....	46
\@gls@noaccess .....	222
\@gls@nohyperlist .....	13
\@gls@onlypremakeg .....	19
\@gls@pl@ .....	231
\@gls@renewglossary .....	127
\@gls@sanitizedesc .....	14
\@gls@sanitizename .....	14
\@gls@sanitizesort .....	14
\@gls@sanitizesymbol .....	14
\@gls@saveentrycounter .....	59
\@gls@setcounter .....	41
\@gls@setupsort@def .....	10
\@gls@setupsort@standard .....	9
\@gls@setupsort@use .....	10
\@gls@tmpb .....	63
\@gls@toc .....	28
\@gls@updatechecked .....	63
\@gls@xdy@Lclass@Alpha-page-numbers	
.....	33
\@gls@xdy@Lclass@Appendix-page-numbers	
.....	33
\@gls@xdy@Lclass@Roman-page-numbers	
.....	33
\@gls@xdy@Lclass@alpha-page-numbers	
.....	33
\@gls@xdy@Lclass@arabic-page-numbers	
.....	33

\@gls@xdy@Lclass@arabic-section-numbers	A	
..... 33	\Ac	163
\@gls@xdy@Lclass@roman-page-numbers	\ac	163
..... 32	access (key)	221
\@gls@xdy@locationlist	accsupp package	220
..... 32	\accsuppglossaryentryfield	237
\@gls@xdy@checkbackslash	\accsuppglossarysubentryfield	237
..... 67	\Acf	163
\@gls@xdy@checkquote	\acf	163
..... 67	\Acfp	163
\@gls@Alphacompositor	\acfp	163
..... 24, 33	\Acl	163
\@gls@sacronymlists	\acl	163
..... 12	\Aclp	163
\@gls@defaultplural	\acp	164
..... 46	\acrfootnote	151
\@gls@defaultsort	\ACRfull	148
..... 46	\Acrfull	147
\@glsdisp	\acrfull	147
..... 78	\acrfullformat	147
\@gls@firstletter	\ACRfullpl	149
..... 117	\Acrfullpl	149
\@gls@hypernumber	\acrfullpl	148
..... 143	\acrlinkfootnote	151
\@glslink	\acrlinkfullformat	147
..... 68	\ACRlong	106
\@gls@minrange	\Acrlong	106
..... 117	\acrlong	105
\@gls@nextpages	\ACRlongpl	108
..... 136	\Acrlongpl	108
\@gls@noname	\acrlongpl	107
..... 46	\acrnameformat	149, 155
\@gls@nonextpages	\acrnolinkfootnote	151
..... 136	acronym (option)	12
\@gls@openfile	\acronymfont	149, 153, 156, 157, 159
..... 124	acronymlists (option)	13
\@gls@order	\acronymname	20
..... 17	\acronymtype	11, 145, 146
\@gls@pl@	\acrpluralsuffix	146
..... 74	\ACRshort	103
\@gls@target	\Acrshort	102
..... 68	\acrshort	102
\@gls@widestname	\ACRshortpl	105
..... 211	\Acrshortpl	104
\@ist@filename	\acrshortpl	103
..... 23	\Acs	162
\@makeglossary	\acs	162
..... 123		
\@newglossary		
..... 41		
\@newglossaryentryposthook		
..... 51		
\@newglossaryentryprehook		
..... 51		
\@nopostdesc		
..... 23		
\@onlypremakeg		
..... 19		
\@p@glossarysection		
..... 27		
\@set@glo@numformat		
..... 61, 216		
\@sgls		
..... 69, 78		
\@sgls@link		
..... 58		
\@wrglossary		
..... 127		
\@xdy@main@language		
..... 17		
\@xdy@attributelist		
..... 29		
\@xdy@attributes		
..... 29		
\@xdy@language		
..... 36		
\@xdy@lettergroups		
..... 37		
\@xdy@locationclassorder		
..... 34		
\@xdy@locref		
..... 29		
\@xdy@requiredstyles		
..... 35		
\@xdy@sortrules		
..... 35		
\@xdy@useralphabets		
..... 31		
\@xdy@userlocationdefs		
..... 33		
\@xdy@userlocationnames		
..... 33		

<code>\Acsp</code> .....	163	<code>\defglssdisplay</code> ....	41, 42, 44, 56, 57
<code>\acsp</code> .....	162	<code>\defglssdisplayfirst</code>	41, 42, 44, 56, 57
<code>\addglossarytocaptions</code> .....	21	<code>\DefineAcronymSynonyms</code> .....	162
<code>\addto</code> .....	21	<code>\delimN</code> .....	25, 143
<code>align (environment)</code> .....	59, 60	<code>\delimR</code> .....	25, 143
<code>altlist (style)</code> .....	177	<code>description (environment)</code> .....	176
<code>altlistgroup (style)</code> .....	178	<code>description (key)</code> .....	42
<code>altlisthypergroup (style)</code> .....	178	<code>description (option)</code> .....	16
<code>atlong4col (style)</code> .....	184	<code>descriptionaccess (key)</code> .....	222
<code>atlong4colborder (style)</code> .....	184	<code>\DescriptionDUANewAcronymDef</code>	153
<code>atlong4colheader (style)</code> .....	184	<code>\DescriptionFootnoteNewAcronymDef</code>	152, 238
<code>atlong4colheaderborder (style)</code>	185	<code>\descriptionname</code> .....	20
<code>atlongragged4col (style)</code> .....	188	<code>\DescriptionNewAcronymDef</code>	155, 238
<code>atlongragged4colborder (style)</code>	189	<code>descriptionplural (key)</code> .....	43
<code>atlongragged4colheader (style)</code>	189	<code>descriptionpluralaccess (key)</code> .	222
<code>atlongragged4colheaderborder</code>		<code>doc package</code> .....	4, 11
(style) .....	190	<code>dua (option)</code> .....	16
<code>\altnewglossary</code> .....	41	<code>\DUANewAcronymDef</code> .....	159
<code>altsuper4col (style)</code> .....	199		
<code>altsuper4colborder (style)</code> .....	199	<b>E</b>	
<code>altsuper4colheader (style)</code> .....	199	<code>entrycounter (option)</code> .....	8
<code>altsuper4colheaderborder (style)</code>	200	<code>entrycounterwithin (option)</code> .....	8
<code>altsuperragged4col (style)</code> .....	204	<code>\entryname</code> .....	20
<code>altsuperragged4colborder (style)</code>	205	<code>environments:</code>	
<code>altsuperragged4colheader (style)</code>	205	<code>align</code> .....	59, 60
<code>altsuperragged4colheaderborder</code>		<code>description</code> .....	176
(style) .....	205	<code>longtable</code> .....	7, 164, 179–190
<code>alttree (style)</code> .....	211	<code>multicols</code> .....	190
<code>alttreegroup (style)</code> .....	213	<code>supertabular</code> ....	7, 164, 194–205
<code>alttreehypergroup (style)</code> .....	213	<code>theglossary</code> .....	4,
<code>amsgen package</code> .....	4, 57	5, 25, 139, 142, 192, 193, 208–211	
<code>\andname</code> .....	21	<code>theindex</code> .....	206
<code>array package</code> .....	185, 200	<code>equation counter</code> .....	59, 60
<code>article class</code> .....	130	<code>etoolbox package</code> .....	4, 170
<b>B</b>		<b>F</b>	
<code>babel package</code> .....	18–21, 36, 242	<code>file types</code>	
		<code>.aux</code> .....	134
<b>C</b>		<code>.glo</code> .....	52
<code>\capitalisewords</code> .....	171	<code>.ist</code> .....	116, 123
<code>\changes</code> .....	4	<code>.toc</code> .....	28
<code>compatible-2.07 (option)</code> .....	18	<code>.xdy</code> .....	23
<code>counter (key)</code> .....	44	<code>glo</code> .....	169
<code>counter (option)</code> .....	13	<code>\findrootlanguage</code> .....	36
<code>\CustomAcronymFields</code> .....	161	<code>first (key)</code> .....	43
<code>\CustomNewAcronymDef</code> .....	161	<code>firstaccess (key)</code> .....	221
		<code>\firstacronymfont</code> .....	149
<b>D</b>		<code>firstplural (key)</code> .....	43
<code>\DeclareAcronymList</code> .....	12	<code>firstpluralaccess (key)</code> .....	221
<code>\DefaultNewAcronymDef</code> ...	150, 237		

footnote (option) .....	16	textaccess .....	221
\FootnoteNewAcronymDef ..	156, 239	type .....	44
\forallallglossaries .....	37	user1 .....	45
\forallallglsentries .....	38	user2 .....	45
\forallorglsentries .....	37	user3 .....	45
<b>G</b>			
\glolinkprefix .....	59	user4 .....	45
glossareentry counter .....	137	user5 .....	45
glossaries package .....		user6 .....	45
.....	36, 117, 164, 176, 214, 220	glossary package .....	1, 145
glossaries-accsupp package ....	52, 220	glossary styles:	
\GlossariesWarning .....	18	altlist .....	177, 178
\GlossariesWarningNoLine ....	18	altlist .....	177
\glossary .....	40, 123, 126, 141	altlistgroup .....	178
glossary counters:		altlistgroup .....	178
glossaryentry .....	137	altlisthypergroup .....	178
glossarysubentry .....	137	altlisthypergroup .....	178
glossary keys:		altlong4col .....	184, 188
access .....	221	altlong4col .....	184
counter .....	44	altlong4colborder .....	184
description .....	42	altlong4colborder .....	184
descriptionaccess .....	222	altlong4colheader .....	184
descriptionplural .....	43	altlong4colheader .....	184
descriptionpluralaccess ..	222	altlong4colheaderborder ..	185
first .....	43	altlong4colheaderborder ..	185
firstaccess .....	221	altlongragged4col ....	188, 189
firstplural .....	43	altlongragged4col .....	188
firstpluralaccess .....	221	altlongragged4colborder ..	189
long .....	46	altlongragged4colborder ..	189
longaccess .....	222	altlongragged4colheader ..	189
longplural .....	46	altlongragged4colheader ..	189
longpluralaccess .....	222	altlongragged4colheaderborder	
name .....	42	.....	190
nonumberlist .....	44	altlongragged4colheaderborder	
parent .....	44	.....	190
plural .....	43	altsuper4col .....	199, 200, 204
pluralaccess .....	221	altsuper4col .....	199
see .....	44	altsuper4colborder .....	199
short .....	45	altsuper4colborder .....	199
shortaccess .....	222	altsuper4colheader .....	199
shortplural .....	45	altsuper4colheader .....	199
shortpluralaccess .....	222	altsuper4colheaderborder .	200
sort .....	43	altsuper4colheaderborder .	200
symbol .....	43	altsuperragged4col ...	204, 205
symbolaccess .....	221	altsuperragged4col .....	204
symbolplural .....	44	altsuperragged4colborder .	205
symbolpluralaccess .....	221	altsuperragged4colborder .	205
text .....	43	altsuperragged4colheader .	205
		altsuperragged4colheader .	205

altsuperragged4colheaderborder	longragged	185–187
..... 205	longragged	185
altsuperragged4colheaderborder	longragged3col	187, 188
..... 205	longragged3col	187
almtree	longragged3colborder	187
..... 193, 210, 211, 213	longragged3colborder	187
almtree	longragged3colheader	188
..... 211	longragged3colheader	188
almtreegroup	longragged3colheaderborder	188
..... 213	longragged3colheaderborder	188
almtreegroup	longraggedborder	186
..... 213	longraggedborder	186
almtreehypergroup	longraggedheader	186
..... 213	longraggedheader	186
almtreehypergroup	longraggedheaderborder	186
..... 213	longraggedheaderborder	186
index	mcotalmtree	193
..... 190, 206–208	mcotalmtree	193
index	mcotalmtreegroup	193
..... 206	mcotalmtreegroup	193
indexgroup	mcotalmtreehypergroup	193
..... 207	mcotalmtreehypergroup	193
indexgroup	mcotindex	191
..... 207	mcotindex	190
indexhypergroup	mcotindexgroup	191
..... 207	mcotindexgroup	191
indexhypergroup	mcotindexhypergroup	191
..... 207	mcotindexhypergroup	191
inline	mcotree	191
..... 174	mcotree	191
list	mcotreegroup	191
..... 6, 176, 177, 179	mcotreehypergroup	192
list	mcotreehypergroup	192
..... 176	mcotreenoname	192
listdotted	mcotreenoname	192
..... 178, 179	mcotreenoname	192
listdotted	mcotreenonamegroup	192
..... 178	mcotreenonamegroup	192
listgroup	mcotreenonamehypergroup	192
..... 177	mcotreenonamehypergroup	192
listgroup	sublistdotted	179
..... 177	super	194–196, 202
listhypergroup	super	194
..... 177	super3col	196, 197
listhypergroup	super3col	196
..... 177	super3colborder	196
long	super3colborder	196
..... 180, 181, 185	super3colheader	197
long	super3colheader	197
..... 180	super3colheaderborder	197
long3col		
..... 181, 182		
long3col		
..... 181		
long3colborder		
..... 181		
long3colborder		
..... 181		
long3colheader		
..... 182		
long3colheader		
..... 182		
long3colheaderborder		
..... 182		
long3colheaderborder		
..... 182		
long4col		
..... 182–184		
long4col		
..... 182		
long4colborder		
..... 183		
long4colborder		
..... 183		
long4colheader		
..... 183		
long4colheader		
..... 183		
long4colheaderborder		
..... 183		
long4colheaderborder		
..... 183		
longborder		
..... 180		
longborder		
..... 180		
longheader		
..... 180		
longheader		
..... 180		
longheaderborder		
..... 181		
longheaderborder		
..... 181		

super3colheaderborder	197	glossary-mcols package	190
super4col	197–199	glossary-super package	
super4col	197		7, 179, 194, 200, 204
super4colborder	198	glossary-superragged package	200
super4colborder	198	glossary-tree package	8, 206
super4colheader	198	glossaryentry (counter)	137
super4colheader	198	glossaryentry counter	8, 138, 139
super4colheaderborder	199	\glossaryentryfield	43, 140, 142
super4colheaderborder	199	\glossaryentrynumber	136
superborder	195	\glossaryentrynumbers	6, 25, 133, 135
superborder	195	\glossaryheader	139, 142
superheader	195	\glossarymark	8, 26
superheader	195	\glossaryname	20, 21
superheaderborder	195	\glossarypostamble	26, 142
superheaderborder	195	\glossarypreamble	25, 142
superragged	201, 202	\glossarysection	5, 26, 40
superragged	201	\glossarystyle	141, 164
superragged3col	202–204	glossarysubentry (counter)	137
superragged3col	202	glossarysubentry counter	9, 137–139
superragged3colborder	203	\GLS	72
superragged3colborder	203	\Gls	71, 75, 170
superragged3colheader	203	\gls	4, 43, 55, 56, 58, 69, 72, 74, 79, 81, 82, 84, 85, 87, 88, 90, 91, 93, 94, 96, 97, 99, 100, 138
superragged3colheader	203	\gls@Alphpage	128
superragged3colheaderborder	204	\gls@alphpage	128
superraggedborder	201	\gls@checkisacronymlist	13
superraggedborder	201	\gls@codepage	17
superraggedheader	202	\gls@disablepagerefexpansion	127
superraggedheader	202	\gls@doclearpage	28
superraggedheaderborder	202	\gls@hypergroup prerun	173
superraggedheaderborder	202	\gls@level	46
superraggedright3colheaderborder	204	\gls@numberpage	128
tree	191, 208, 209, 211	\gls@protected@pagefmts	127
tree	208	\gls@Romanpage	128
treegroup	192, 209	\gls@romanpage	128
treegroup	208	\gls@save@numberlist	132
treehypergroup	209	\gls@suffixF	24
treehypergroup	209	\gls@suffixFF	24
treenoname	192, 209, 210	\gls@accessdisplay	228
treenoname	209	\gls@sacccsupp	225
treenonamegroup	210	\gls@sadd	55, 115, 141
treenonamegroup	210	\gls@sadd options	
treenonamehypergroup	210	counter	115
treenonamehypergroup	210	format	115, 142
glossary-hypernav package	116	\gls@saddall	55, 116
glossary-list package	6, 7, 176	\gls@saddall options	
glossary-long package	7, 179, 188, 194	types	115, 116
glossary-longragged package	185	\GlsAddLetterGroup	37

<code>\GlsAddSortRule</code> .....	35	<code>\Glsentryfullpl</code> .....	114
<code>\GlsAddXdyAlphabet</code> .....	32	<code>\glentryfullpl</code> .....	113
<code>\GlsAddXdyAttribute</code> .....	30, 214	<code>\glentryitem</code> .....	139
<code>\GlsAddXdyCounters</code> .....	29, 214	<code>\Glsentrylong</code> .....	113
<code>\GlsAddXdyLocation</code> .....	34, 215	<code>\glentrylong</code> .....	113
<code>\GlsAddXdyStyle</code> .....	35	<code>\glentrylongaccess</code> .....	225
<code>\glsclearpage</code> .....	6	<code>\Glsentrylongpl</code> .....	113
<code>\glsclearpage</code> .....	28	<code>\glentrylongpl</code> .....	113
<code>\glsclosebrace</code> .....	116	<code>\glentrylongpluralaccess</code> ..	225
<code>\glscpositor</code> .....	24, 33	<code>\Glsentryname</code> .....	109
<code>\glscounter</code> .....	13, 41	<code>\glentryname</code> .....	109, 132
<code>\GlsDeclareNoHyperList</code> .....	13	<code>\glentrynumberlist</code> .....	114
<code>\glscdefaulttype</code> .....	11	<code>\Glsentryplural</code> .....	110
<code>\glscdefmain</code> .....	11	<code>\glentryplural</code> .....	110
<code>\GLSdesc</code> .....	88	<code>\glentrypluralaccess</code> .....	224
<code>\Glsdesc</code> .....	87	<code>\Glsentryshort</code> .....	113
<code>\glscdesc</code> .....	87, 88	<code>\glentryshort</code> .....	112
<code>\GLSdescplural</code> .....	89	<code>\glentryshortaccess</code> .....	225
<code>\Glsdescplural</code> .....	89	<code>\Glsentryshortpl</code> .....	113
<code>\glscdescplural</code> .....	88, 89	<code>\glentryshortpl</code> .....	113
<code>\glscdescriptionaccessdisplay</code>	227	<code>\glentryshortpluralaccess</code> ..	225
<code>\glscdescriptionpluralaccessdisplay</code>	227	<code>\glentrysort</code> .....	111
<code>\glscdescwidth</code> ....	179, 185, 194, 200	<code>\Glsentrysymbol</code> .....	110
<code>\glscdisablehyper</code> .....	69	<code>\glentrysymbol</code> .....	110
<code>\glscdisp</code> .....	78	<code>\glentrysymbolaccess</code> .....	224
<code>\glscdisplay</code> .....	14, 42, 43, 56, 69	<code>\Glsentrysymbolplural</code> .....	111
<code>\glscdisplayfirst</code> ..	42, 44, 56, 57, 69	<code>\glentrysymbolplural</code> .....	110
<code>\glscdisplaynumberlist</code> .....	114	<code>\glentrysymbolpluralaccess</code> ..	224
<code>\glscdoifexists</code> .....	38	<code>\Glsentrytext</code> .....	110
<code>\glscdoifnoexists</code> .....	39	<code>\glentrytext</code> .....	110, 132
<code>\glscenablehyper</code> .....	69	<code>\glentrytextaccess</code> .....	224
<code>\glscentryaccess</code> .....	224	<code>\glscentrytype</code> .....	111
<code>\glscentrycounter</code> .....	59	<code>\Glsentryuseri</code> .....	111
<code>\glscentrycounterlabel</code> .....	138	<code>\glscentryuseri</code> .....	111
<code>\Glsentrydesc</code> .....	109	<code>\Glsentryuserii</code> .....	112
<code>\glscentrydesc</code> .....	14, 109	<code>\glscentryuserii</code> .....	112
<code>\glscentrydescaccess</code> .....	224	<code>\Glsentryuseriii</code> .....	112
<code>\Glsentrydescplural</code> .....	110	<code>\glscentryuseriii</code> .....	112
<code>\glscentrydescplural</code> .....	110	<code>\Glsentryuseriv</code> .....	112
<code>\glscentrydescpluralaccess</code> ..	225	<code>\glscentryuseriv</code> .....	112
<code>\Glsentryfirst</code> .....	111	<code>\Glsentryuserv</code> .....	112
<code>\glscentryfirst</code> .....	111	<code>\glscentryuserv</code> .....	112
<code>\glscentryfirstaccess</code> .....	224	<code>\Glsentryuservi</code> .....	112
<code>\Glsentryfirstplural</code> .....	111	<code>\glscentryuservi</code> .....	112
<code>\glscentryfirstplural</code> .....	111	<code>\GLSfirst</code> .....	82
<code>\glscentryfirstpluralaccess</code> ..	224	<code>\Glsfirst</code> .....	81, 82
<code>\Glsentryfull</code> .....	113	<code>\glscfirst</code> .....	81
<code>\glscentryfull</code> .....	113	<code>\glscfirstaccessdisplay</code> .....	226
		<code>\GLSfirstplural</code> .....	85

<code>\Glsfirstplural</code> .....	84	<code>\glsnamefont</code> .....	142
<code>\glsfirstplural</code> .....	84, 85	<code>\glsnavhyperlink</code> .....	172
<code>\glsfirstpluralaccessdisplay</code> .....	226	<code>\glsnavhypertarget</code> .....	172
<code>\glsgetgrouplabel</code> .....	141	<code>\glsnavigation</code> .....	173
<code>\glsgetgrouptitle</code> .....	116, 141	<code>\glsnextpages</code> .....	137
<code>\glsgroupheading</code> .....	140, 142	<code>\glsnonextpages</code> .....	137
<code>\glsgroupskip</code> .....	140, 142, 176	<code>\glsnoxindywarning</code> .....	28
<code>\glshyperlink</code> .....	115	<code>\glsnumberformat</code> .....	25
<code>\glshypernavsep</code> .....	174	<code>\glsnumbersgroupname</code> ..	21, 116, 141
<code>\glshypernumber</code> .....	25, 143	<code>\glsnumlistlastsep</code> .....	115
<code>\glsIfListOfAcronyms</code> .....	12	<code>\glsnumlistsep</code> .....	115
<code>\glsinlinedescformat</code> .....	176	<code>\glsopenbrace</code> .....	116
<code>\glsinlinedopostchild</code> ...	174, 175	<code>\glsorder</code> .....	17
<code>\glsinlineemptydescformat</code> ...	176	<code>\glsorg@endtheglossary</code> .....	5
<code>\glsinlinenameformat</code> .....	176	<code>\glsorg@glossary</code> .....	4
<code>\glsinlineparentchildseparator</code> .....	176	<code>\glsorg@theglossary</code> .....	5
<code>\glsinlinepostchild</code> .....	176	<code>\glsorg@wrglossary</code> .....	4
<code>\glsinlineseparator</code> .....	175	<code>\glspagelistwidth</code> .....	180, 185, 194, 201
<code>\glsinlinesubdescformat</code> .....	176	<code>\glspar</code> .....	23
<code>\glsinlinesubnameformat</code> .....	176	<code>\GLSpl</code> .....	77
<code>\glsinlinesubseparator</code> .....	175	<code>\Glspl</code> .....	75, 170
<code>\glskeylisttok</code> .....	150	<code>\glspl</code> .....	55, 56, 74–76
<code>\glslabeltok</code> .....	150	<code>\GLSplural</code> .....	83
<code>\glslink</code> ..	55, 56, 58, 69, 115, 141, 143	<code>\Glsplural</code> .....	83
<code>\glslink options</code> counter .....	57, 69, 169	<code>\glsplural</code> .....	82, 83
format .....	57, 69, 142	<code>\glspluralaccessdisplay</code> .....	226
hyper .....	58, 69	<code>\glspluralsuffix</code> .....	21, 43
local .....	58	<code>\glspostdescription</code> .....	8
<code>\glslistdottedwidth</code> .....	179	<code>\glspostinline</code> .....	176
<code>\glslocalreset</code> .....	54	<code>\glsquote</code> .....	117
<code>\glslocalresetall</code> .....	55	<code>\glsrefentry</code> .....	138
<code>\glslocalunset</code> .....	54	<code>\glsreset</code> .....	54
<code>\glslocalunsetall</code> .....	55	<code>\glsresetall</code> .....	55
<code>\glslongaccessdisplay</code> .....	228	<code>\glsresetentrylist</code> .....	137
<code>\glslongaccesskey</code> .....	240	<code>\glsresetsubentrycounter</code> .....	137
<code>\glslongkey</code> .....	147	<code>\glssee</code> .....	131
<code>\glslongpluralaccessdisplay</code> ..	228	<code>\glsseeformat</code> .....	119, 131
<code>\glslongpluralaccesskey</code> .....	240	<code>\glsseeitem</code> .....	132
<code>\glslongpluralkey</code> .....	147	<code>\glsseeitemformat</code> .....	132
<code>\glslongtok</code> .....	150	<code>\glsseelastsep</code> .....	132
<code>\glsmakefirsttuc</code> .....	171	<code>\glsseelist</code> .....	131
<code>\glsmcols</code> .....	190	<code>\glsseesep</code> .....	132
<code>\glsmoveentry</code> .....	52	<code>\glsSetAlphaCompositor</code> .....	24
<code>\GLSname</code> .....	86	<code>\glsSetCompositor</code> .....	23, 24
<code>\Glsname</code> .....	86	<code>\glsSetSuffixF</code> .....	24
<code>\glsname</code> .....	85, 86	<code>\glsSetSuffixFF</code> .....	25
<code>\glsnameaccessdisplay</code> .....	225	<code>\glssettoctitle</code> .....	20
		<code>\glssetwidest</code> .....	210
		<code>\GlsSetXdyCodePage</code> .....	36



listhypergroup (style) .....	177	mcolindex (style) .....	190
\loadglsentries .....	11, 55	mcolindexgroup (style) .....	191
long (key) .....	46	mcolindexhypergroup (style) ....	191
long (style) .....	180	mcoltree (style) .....	191
long3col (style) .....	181	mcoltreegroup (style) .....	191
long3colborder (style) .....	181	mcoltreehypergroup (style) ....	192
long3colheader (style) .....	182	mcoltreenoname (style) .....	192
long3colheaderborder (style) ...	182	mcoltreenonamegroup (style) ....	192
long4col (style) .....	182	mcoltreenonamehypergroup (style)	192
long4colborder (style) .....	183	memoir class .....	126
long4colheader (style) .....	183	mfistuc package .....	1
long4colheaderborder (style) ...	183	multicol package .....	190
longaccess (key) .....	222	multicols (environment) .....	190
longborder (style) .....	180		
longheader (style) .....	180	N	
longheaderborder (style) .....	181	name (key) .....	42
longplural (key) .....	46	\newacronym 16, 45, 46, 55, 56, 145, 146	
longpluralaccess (key) .....	222	\newacronymhook .....	150
longragged (style) .....	185	\newglossary . 13, 40, 41, 123, 124, 135	
longragged3col (style) .....	187	\newglossaryentry 14, 46, 55, 56, 146	
longragged3colborder (style) ...	187	\newglossaryentry options	
longragged3colheader (style) ...	188	access .....	222, 224
longragged3colheaderborder		counter .....	44
(style) .....	188	description .....	14, 16,
longraggedbborder (style) .....	186	42, 46, 47, 56, 87, 109, 146, 157, 222	
longraggedheader (style) .....	186	descriptionaccess .....	224, 227
longraggedheaderborder (style) .	186	descriptionplural .....	88, 222
longtable (environment) .....		descriptionpluralaccess ...	225, 227
.....	7, 164, 179–190	first .....	43, 49,
longtable package .....	179, 185	56, 69, 81, 111, 155, 158, 159, 221	
M		firstaccess .....	224, 226
\makefirstuc .....	170	firstplural ....	43, 49, 56, 84, 111, 221
makeglossaries 17, 23, 36, 40, 124, 134		firstpluralaccess .....	224, 226
\makeglossaries .....		format .....	118
..... 19, 23, 24, 40, 41, 124, 125		long .....	113, 222
\makeglossary .....	125	longaccess .....	225, 228
makeindex .....	255	longplural .....	113, 222
makeindex .....	17, 21,	longpluralaccess .....	225, 228
23–25, 40–43, 53, 61, 64, 116,		name .....	14,
119, 121, 123, 129, 140, 215, 216		15, 42, 43, 46, 47, 85, 109, 132, 221	
delim_n .....	25	nonumberlist .....	44
delim_r .....	25	parent .....	44, 46
page_compositor .....	23	plural .....	43, 49, 56, 82, 221
special characters .....	62, 63, 116	pluralaccess .....	224, 226
\MakeUppercase .....	8	see .....	7, 44
mcolalmtree (style) .....	193	short .....	112, 222
mcolalmtreegroup (style) .....	193	shortaccess .....	225, 227
mcolalmtreehypergroup (style) ..	193	shortplural .....	113, 222
		shortpluralaccess .....	225, 227

sort .....	14, 15, 43, 111, 140	entrycounterwithin .....	8
symbol 14, 15, 42, 43, 56, 90, 110,		footnote .....	70,
152–155, 158, 182, 197, 221, 222		72–74, 76, 77, 79, 153–156, 229–234	
symbolaccess .....	224, 226	footnote .....	16
symbolplural .....	91, 221	hyperfirst	
symbolpluralaccess .....	224, 227	false .....	70, 72–74, 76, 77, 79
text 43, 56, 69, 79, 110, 152, 155, 221		hyperfirst .....	16
textaccess .....	224, 226	indexonlyfirst .....	262
type .....	11, 44, 55, 111	indexonlyfirst .....	16
user1 .....	93, 111, 222	makeindex .....	119, 169
user2 .....	94, 112	nogroupskip .....	8
user3 .....	96, 112	nohypertypes .....	14
user4 .....	97, 112	nolist .....	164
user5 .....	99, 112	nolist .....	7
user6 .....	100, 112, 222	nolong .....	164, 179
\newglossarystyle .....	142	nolong .....	7
nogroupskip (option) .....	8	nomain .....	11
nohypertypes (option) .....	14	nonumberlist .....	7
\noist .....	123, 169, 220	nonumberlist .....	7
nolist (option) .....	7	nopostdot .....	8
nolong (option) .....	7	nostyles .....	8
nonumberlist (key) .....	44	nosuper .....	164
nonumberlist (option) .....	7	nosuper .....	7
\nopostdesc .....	23	notree .....	164
nopostdot (option) .....	8	notree .....	8
nostyles (option) .....	8	numberedsection .....	6
nosuper (option) .....	7	numberline .....	5
notree (option) .....	8	numberline .....	5
numberedsection (option) .....	6	order .....	17
numberline (option) .....	5	sanitize .....	14, 15, 42, 109, 110
<b>O</b>		sanitize .....	15
\oldacronym .....	145	savenumberlist .....	7
order (option) .....	17	savewrites .....	18, 260
<b>P</b>		false .....	123
package options:		true .....	125
acronym ....	11, 12, 20, 135, 145, 146	savewrites .....	18
acronym .....	12	section .....	5, 27
acronymlists .....	13	section .....	5
compatible-2.07 .....	18	seeautonumberlist .....	7
counter .....	13	shotcuts .....	17
counter .....	13	smallcaps .....	16
description .....	155, 156	smaller .....	16
description .....	16	sort	
dua .....	154–156	def .....	9
dua .....	16	standard .....	9
entrycounter .....	137	use .....	9
true .....	8	sort .....	9
entrycounter .....	8	style .....	6, 164
		style .....	6



<code>\showglossaryentries</code> .....	169	<code>supertabular</code> package ...	7, 164, 194, 200
<code>\showglossaryin</code> .....	168	<code>symbol</code> (key) .....	43
<code>\showglossaryout</code> .....	168	<code>symbolaccess</code> (key) .....	221
<code>\showglossarytitle</code> .....	169	<code>\symbolname</code> .....	20
<code>\showglosymbol</code> .....	167	<code>symbolplural</code> (key) .....	44
<code>\showglosymbolaccess</code> .....	241	<code>symbolpluralaccess</code> (key) .....	221
<code>\showglosymbolplural</code> .....	167		
<code>\showglosymbolpluralaccess</code> ..	241	<b>T</b>	
<code>\showglotext</code> .....	165	<code>text</code> (key) .....	43
<code>\showglotextaccess</code> .....	241	<code>textaccess</code> (key) .....	221
<code>\showglotype</code> .....	165	<code>\theequation</code> .....	130
<code>\showglouseri</code> .....	166	<code>theglossary</code> (environment) ...	4, 5, 25, 139, 142, 192, 193, 208–211
<code>\showglouserii</code> .....	166	<code>\theHequation</code> .....	130
<code>\showglouseriii</code> .....	166	<code>theindex</code> (environment) .....	206
<code>\showglouseriv</code> .....	166	<code>toc</code> (option) .....	5
<code>\showglouserv</code> .....	166	<code>\translate</code> .....	21
<code>\showglouservi</code> .....	167	<code>translate</code> (option) .....	15
<code>smallcaps</code> (option) .....	16	<code>translator</code> package .....	
<code>smaller</code> (option) .....	16	.....	18, 21, 132, 242, 251–254
<code>\SmallNewAcronymDef</code> .....	158, 239	<code>tree</code> (style) .....	208
<code>sort</code> (key) .....	43	<code>treegroup</code> (style) .....	208
<code>sort</code> (option) .....	9	<code>treehypergroup</code> (style) .....	209
<code>style</code> (option) .....	6	<code>treenoname</code> (style) .....	209
<code>subentrycounter</code> (option) .....	9	<code>treenonamegroup</code> (style) .....	210
<code>\subitem</code> .....	206	<code>treenonamehypergroup</code> (style) ...	210
<code>sublistdotted</code> (style) .....	179	<code>type</code> (key) .....	44
<code>\subsubitem</code> .....	206		
<code>super</code> (style) .....	194	<b>U</b>	
<code>super3col</code> (style) .....	196	<code>ucmark</code> (option) .....	8
<code>super3colborder</code> (style) .....	196	<code>user1</code> (key) .....	45
<code>super3colheader</code> (style) .....	197	<code>user2</code> (key) .....	45
<code>super3colheaderborder</code> (style) ..	197	<code>user3</code> (key) .....	45
<code>super4col</code> (style) .....	197	<code>user4</code> (key) .....	45
<code>super4colborder</code> (style) .....	198	<code>user5</code> (key) .....	45
<code>super4colheader</code> (style) .....	198	<code>user6</code> (key) .....	45
<code>super4colheaderborder</code> (style) ..	199		
<code>superborder</code> (style) .....	195	<b>W</b>	
<code>superheader</code> (style) .....	195	<code>\warn@nomakeglossaries</code> .....	124
<code>superheaderborder</code> (style) .....	195	<code>\warn@noprintglossary</code> .....	132
<code>superragged</code> (style) .....	201	<code>\writeist</code> ..	23, 30, 31, 34, 117, 214, 216
<code>superragged3col</code> (style) .....	202		
<code>superragged3colborder</code> (style) ..	203	<b>X</b>	
<code>superragged3colheader</code> (style) ..	203	<code>\xcapitalisewords</code> .....	172
<code>superraggedborder</code> (style) .....	201	<code>\xglsaccsupp</code> .....	225
<code>superraggedheader</code> (style) .....	202	<code>xindy</code> .....	255
<code>superraggedheaderborder</code> (style)	202	<code>xindy</code> .....	17, 18, 23, 24, 28, 32, 34–37, 52, 53, 67, 117, 119, 129, 134, 140, 169, 215, 216
<code>superraggedright3colheaderborder</code>		<code>\xmakefirstuc</code> .....	171
(style) .....	204	<code>xspace</code> package .....	4, 145, 146
<code>supertabular</code> (environment) ....			
.....	7, 164, 194–205		